Journal of Computer Science 6 (2): 141-162, 2010
ISSN 1549-3636
© 2010 Science Publications

Soft Computing in Optimizing Assembly Lines Balancing

[1]Muhammad Zaini Matondang and [1,2]Muhammad Ikhwan Jambak
[1]Department of Modeling and Industrial Computing,
Faculty of Computer Science and Information System,
[2]Soft Computing Research Group, Research Alliance on K-Economy,
University Technology Malaysia, 81310 Johor Bahru, Malaysia

Abstract: As part of manufacturing systems, the assembly line has become one of the most valuable researches to accomplish the real world problems related to them. Many efforts have been made to seek the best techniques in optimizing assembly lines. **Problem statement:** Since it was published by Salveson in 1955, some methods and techniques have been developed based on mathematical modeling. In recent years, some researches in Assembly Line Balancing (ALB) have been conducted using Soft Computing (SC) approaches. However, there is no comprehensive survey studies conducted regarding the use of SC in ALB problems, which is became the aim of this study. **Approach:** This study reviewed published literatures and previous related works that applied SC in solving ALB problems. Main outcomes: This study looks into the suitability of SC approaches in several types of ALB problems. Furthermore, this study provides the classification of ALB problems that can facilitate distinguishing those problems as fields of research. **Result:** This study found that Genetic Algorithms (GAs) are predominantly applied to solve ALB problems compared to other SC approaches. This high suitability in ALB refers to GAs' main characteristics that include its robustness and flexibility. These SC approaches have mostly been applied to simple ALB problems, which are not problems that are covered in a real complex manufacturing environment. **Conclusion/Recommendations:** This study recommends that future researches in ALB should be conducted with regard to other issues, beyond the simple ALB problems and more practical to the industries. Besides the advantages of GAs, there are still opportunities to use other SC approaches and the hybrid-systems among them that could increase the suitability of these approaches, especially for multi-objective ALB problems. This study also recommends that human involvement in ALB needs to be considered as a problem factor in ALB.

Key words: Assembly lines balancing, soft computing, optimization

INTRODUCTION

A manufacturing system could be defined as a collection of integrated equipment (including production machines and tools, material handling and work-positioning devices and computer systems) and human resources, whose function is to perform one or more processing and/or assembly operations on raw materials, a part, or set of parts (Groover, 2008). In this system, human resources are required either full time or periodically to keep the system running. There are seven systems included in a manufacturing system: they are; taxonomy, single-station cells, group technology, a flexible manufacturing system, manual assembly lines, automated assembly lines and transfer lines. In this study, the discussion will focus on an assembly line system. It is both an old problem and a new problem, due to the fact that many researchers still attempt to stumble on optimized ways, methods or techniques to assembly lines balancing.

Balancing assembly lines becomes one of the most important parts for an industrial manufacturing system that should be supervised carefully. The success of achieving the goal of production is influenced significantly by balancing assembly lines. Since then, many industries and for sure researchers, attempt to find the best methods or techniques to keep the assembly line balanced and even to make it more efficient. Furthermore, this problem is known as an assembly lines balancing problem. As there are many researches that have been performed, few techniques and methods have been used in solving the optimization problems. They are

Corresponding Author: Muhammad Ikhwan Jambak, Department of Modeling and Industrial Computing, Faculty of Computer Science and Information System, University Technology Malaysia, 81310, Johor Bahru, Malaysia

based on mathematical modeling, such as the use of linear programming and then the latest are based on the soft computing approach, with the more famous one being the use of genetic algorithms.

MATERIALS AND METHODS

Optimization could be defined as the effort, way, technique, method or system to use for calculating or finding the best possibilities of utilization of resources (which can be people, time, process, vehicles, equipment, raw materials, supplies and others) needed to achieve an expected result, with it being the best possible solution to the problem. In mathematics, the simplest case of optimization, or mathematical programming, refers to the study of problems in which one seeks to minimize or maximize a real function by systematically choosing the values of real or integer variables from within an allowed set. The first optimization technique, which is known as steepest descent, goes back to Gauss. Historically, the first term to be introduced was ‘linear programming’, which was invented by George Dantzig in the 1940s. The term ‘programming’ in this context does not refer to computer programming (although computers are nowadays used extensively to solve mathematical problems). Instead, the term comes from the use of program by the United States military to refer to proposed training and logistics schedules, which were the problems that Dantzig was studying at the time. The wide variety of applications benefiting from optimization include: Production planning and scheduling, raw material blending, yield and revenue management, crew scheduling, financial portfolio management, product configuration, technician and truck dispatching, satellite mission planning and others. However, for those kinds of applications, there are many techniques and methods used for optimization purpose and basically they are divided according to the number of variables involved, which are called Single Variable Optimization (SVO) and Multi-Variable Optimization (MVO).

Another literature (Chong and Zak, 2008) stated the optimization problems are divided into two twice-differentiable functions: Constrained and unconstrained problems. Unconstrained problems can be solved by finding the points where the gradient of the objective function is zero and using the Hessian matrix to classify the type of each point. The existence of derivatives is not always assumed and many methods were devised for specific situations. The basic classes of methods, based on smoothness of the objective function, are: Combinatorial methods, derivative-free methods, first and second order methods, gradient descent (aka steepest descent or steepest ascent), interior point methods, line search method, Newton's method, quasi-Newton methods, subgradient method-similar to gradient method in case there are no gradients and many others. Constrained problems can often be transformed into unconstrained problems with the help of Lagrange multipliers. Few other popular methods such as ant colony optimization, beam search, bee algorithms, differential evolution, dynamic relaxation, evolution strategy, genetic algorithms, harmony search, hill climbing, particle swarm optimization, quantum annealing, simulated annealing, stochastic tunneling and Tabu search. However, among those methods, genetic algorithms, which are part of soft computing approaches, is the most used technique today for optimization matters, even compared with other soft computing approaches. It is because genetic algorithms provide an alternative to traditional optimization techniques by using directed random searches to locate optimum solutions in complex landscapes.

In this study, a survey study of soft computing applications in assembly line balancing is presented. The survey study focused on the efforts of previous works in finding the best techniques to optimize assembly lines based on soft computing approaches. Furthermore, this study is looking for the suitability of SC approaches in several types of ALB problems.

The discussions of this study are managed as follow. At first, we present an overview of this study and explain a few definitions and facts about optimization problems and the techniques used. We continue the discussion by detailing the assembly lines balancing, including its characteristics, layout, problems classification, and its role in manufacturing. A brief discussion of soft computing and its general capabilities is presented, and furthermore the soft computing applications in assembly lines balancing as the core this survey. At the end of this critical review, we present our conclusion and recommendation for future researches.

Assembly line balancing: we brief an introduction in here to get more understanding about assembly lines balancing. The discussions cover basic knowledge about assembly lines, a few definitions, characteristics, problems classifications and the important of assembly lines balancing.

Assembly lines and the balancing problem: There are three reasons why assembly lines were developed. They are for a cost-efficient mass-production of standardized products, designed to exploit a high specialization of labor and the associated learning effects (Shtub and Dhar-El, 1989). Since then assembly lines have been gradually improved. Henry Ford's introduction of assembly lines, from straight single-model lines to more flexible systems including, among others, lines with parallel work stations or tasks, customer-oriented mixed-model and multi-model lines, U-shaped lines as well as un-paced lines with intermediate buffers (Becker and Scholl, 2006).

Few definitions of assembly lines are given by few researchers. Becker and Scholl (2006) said that assembly lines are a traditional and still effective means of mass and large-scale production. They are also dubbed as flow-oriented production systems which are still typical in the industrial production of high-quantity standardized commodities and even gain importance in low-volume production of customized products. Lusa (2008) said that assembly lines could be defined as a production system made up of a series of workstations that are connected by a conveyor belt or a similar system that transports the object that is being assembled. Furthermore, Yaman (2008) stated that assembly lines are an example of flow lines which is the most commonly used system in a mass-production environment. Assembly lines enable the assembly of complex products by workers who have received a short training period (Gunasekaran and Cecile, 1998). Thus, an efficient assembly line design, as a part of a manufacturing system, is a vital problem for some companies. An assembly line is a usual solution for medium and high-production volumes.

In any case, an important decision problem, called also assembly line balancing problem, arises and has to be solved when (re-) configuring an assembly line. It consists of distributing the total workload for manufacturing any unit of the product to be assembled among the workstations along the line. Falkenauer (2005) explained that Assembly Lines Balancing (ALB), or simply Line Balancing (LB), are the problem of assigning operations to workstations along an assembly line, in such a way that the assignment is optimal in some sense. It has been an optimization problem which was very crucial for many industries. By managing an assembly line, few advantages occur, such as better labor and machine utilization, easy learning for workers, less work-in-process inventory and less space requirement (Veeramani, 2001). Mayers and Stephens (2005) stated some purposes of the assembly lines balancing technique. They are to equalize the workload among the assemblers, to identify any operational bottlenecks, to establish the speed of assembly lines, to determine the number of workstations, to determine the labor cost of assembly and packaging, to establish the percentage workload of each operator, to assist in plan layout and to reduce production costs.

Few literatures have stated the main objective of an assembly line, which is to increase the efficiency of the system by maximizing the ratio between throughput and required cost. An assembly chart shows the sequence of operations required to put a product together as the final stage of manufacture. This assembly process can be shown graphically by using the parts list and related drawings. In complex products, the sequence of assembly may have other alternatives. For a good decision among these alternatives, time-standards and precedence lists are required (Meyers and Stephens, 2005; Boysen *et al*., 2007). With this background, an assembly line may be designed and balanced with the aim of optimizing the assembly system. For other descriptions of assembly systems and different balancing problems one could refer to Buxey *et al*. (1973); Baybars (1986); Shtub and Dhar-El (1989); Gosh and Gagnon (1989); Erel and Sarin (1998); Scholl (1999); Rekiek and Delchambre (2001) and the most recent survey of Becker and Scholl (2006).

Characteristics of assembly lines: There is a work element and workstation as a part in assembly lines. Then, it is better to know about a work element and workstation first, before knowing all about the assembly lines. A work element is the smallest unit productive work that adds values to the product, such as tightening (thinning/reduction) a screw, welding, inserting a gear assembly. A workstation is also dubbed as a collection of a set of work elements that are performed there. A product is passed down the line and visits each workstation in sequence. An assembly line contains of a set of sequential workstations, typically connected by a continuous material handling system. It is designed to assemble component parts of a product and perform any related operations to produce the finished product. There also other components in there, namely workers (manual and robotic), a material handling system (conveyors), buffers, unloading and storage space, layout (linear, U-shape and others).

Referring to Tasan and Tunali (2008), an assembly line consists of a sequence of tasks, each having an operational processing time and a set of precedence relations, is widely adopted in manufacturing plans (by previous literature (Becker and Scholl, 2006), a sequence of workstations have the same meaning with a sequence of tasks in this context). Precedence relations contain the order in which tasks must be performed.

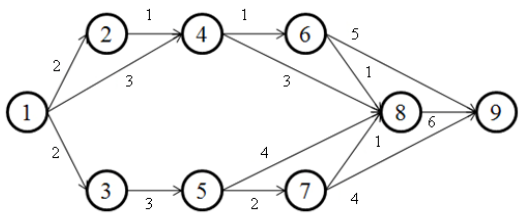


Fig. 1: Precedence graph

Table 1: List of common notations

Notations	Definitions
n	Number of workstations; $i = 1,\ldots,n$
c	Cycle time
m	Number of tasks; $j = 1,\ldots,m$
t_j	Processing time of tasks j
t_{sum}	Total processing time of tasks; $t_{sum} = \sum_{j=1}^{m} t_j$
WS_i	Workstation load of workstation i
$t(WS_i)$	Workstation time of workstation i; $t(WS_i) = \sum_{j \in WS_i} t_j$
$Max(t(WS_i))$	Maximum workstation time
K	Largest single processing time of a task, a constant
N_v	Number of violations in precedence relations
$\tilde{c}$	Fuzzy cycle time
$t(\tilde{W}_i S)$	Fuzzy workstation times for workstation s
n_s	Number of workstations in solution
M	Number of models; $k = 1,\ldots,M$
qk	Demand ratio of model k
it_{ik}	Idle time for workstation i after processing model k
IT_i	Average idle time for workstation i; $IT_i = \sum_{k=1}^{M} qk\, it_{ik}$
E	Line efficiency
$f(s)$	Fitness function of a solution c

Figure 1 illustrates an example of precedence relations by a representation of a precedence graph, which contains 9 nodes for tasks, node weights in italic for task-processing times and arcs for orderings. It is noted that the most commonly used objective function in the literature is the maximization of line efficiency:

$$E = \frac{t_{sum}}{n \cdot c}$$

The following Table 1 presents the widely-used notations in assembly lines balancing literature (Tasan and Tunali, 2008).

As follows, the characterizations of the relevant properties of assembly lines, which have to be considered when balancing those lines, are given:

- Number and variety of products: If only one product or several products with (almost) identical production processes, e.g., production of compact discs (Lebefromm, 1999) or drinking cans (Grabau and Maurer, 1998) are assembled, the production system can be treated as a single-model line.

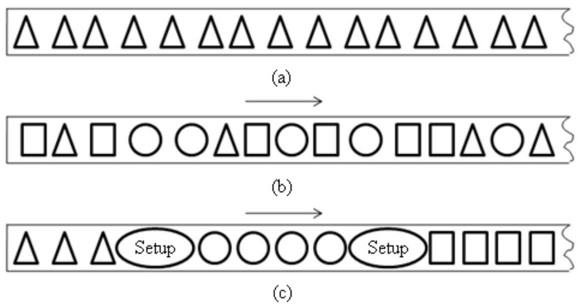


Fig. 2: Assembly line based on the number and variety of the products (redrawn Becker and Scholl, 2006). (a) single-model line; (b) mixed-model line; (c) multi-model line

In modern production systems however, several products or different models of the same base product often share the same assembly line. In general, two different forms of organization are distinguished (c.f. Wild, 1972; Buxey *et al*., 1973): A mixed-model line (A mixed-model line produces the units of different models in an arbitrarily intermixed sequence (cf. Scholl, 1999)) and a multi-model line (A multi-model line produces a sequence of batches (each containing units of only one model or a group of similar models) with intermediate setup operations.). An illustration is given in Fig. 2, which shows the characteristics of assembly lines based on number and variety of the products

- Line control: Assembly systems can be distinguished with regard to the control of job movements between stations. The exact type of line control, which has far-reaching consequences for the structure of the balancing decision, is divided into paced and un-paced lines
- Variability of task times: In reality, task times are basically never deterministic (Tempelmeier, 2003)
- Line layout: Traditionally, an assembly line is organized as a serial line, where single stations are arranged along a (straight) conveyor belt. The actual line layout is, however, not necessarily determined prior to the balancing decision. The real-world arrangement of the conveyor belt does not usually affect the assignment decision and can thus be ignored
- Parallelization of assembly work: Assembly line production makes intensive use of increasing labor efficiency by partitioning the total work among different productive units
- Equipment and processing alternatives: In order to perform a task assigned, the station must be equipped with productive resources like operators,

machines and tools which provide the skills and/or technological capabilities required. Furthermore, the necessary material must be made available

- Assignment restrictions: In ALB, task assignments to stations are always restricted by precedence relationships. In model formulations, the corresponding precedence graph might either have a general structure or be restricted to some special graph type, e.g., linear (Kimms, 2000), diverging or converging graphs. In any case, the precedence graph has to be (made) acyclic to find feasible task-processing sequences (Ahamdi and Wurgaft, 1994)

Objectives: Several of the extensions outlined above can only be considered in a meaningful way, if other objectives than the capacity-oriented ones introduced in previous discussion are observed. Whenever alternative resources are available, resource costs will need to be regarded in the associated selection problem.

Problem classification in assembly lines: There are a few ways in defining the problem of assembly lines balancing, while the problem also has variations which may add some complexity to the problem. Here are some variations which are introduced by Chow (1990):

- Multiple products: Since assembly processes and process times may not be the same for different products, a single line cannot be balanced for all products
- Variable process time: Variability may take a number of different forms and the two most common ones result from human inconsistencies found at manual operations and different reject conditions at test/ inspection operations
- Multiple workstations: If the mean process time of an operation is larger than a planned completion cycle, multiple workstations are needed. However, line sizing and balancing become interrelated problems
- Human factors: According to industrial experiences. it has been shown that the repetition of the same motion pattern induces excessive muscle fatigue and may lead to body injury
- Product characteristics: An assembly line is usually composed of a number of subassemblies and tasks that belong to different subassemblies should not be assigned to the same operation
- Length of cycle time: Assembly lines balancing problems are dependent on the selected cycle time and that the determination of cycle time is a complicated problem

Apart from the variations in the ways in defining the problems of assembly lines balancing as listed above and as stated earlier in this study, assembly lines balancing research has traditionally focused on the Simple Assembly Line Balancing Problem (SALBP), which has some restricting assumptions. However, nowadays, a lot of research work has been done in order to describe and solve more realistic generalized problems, namely General Assembly Lines Balancing Problems (GALBP).

Several version of ALBP also arise by varying the objective function (Scholl, 1999): They are Type-1 until Type-5 and Type-E and Type-F. Type-1 and Type-2 have a dual relationship; the first one tries to minimize the number of workstations for a given cycle time and the second one tries to minimize the cycle time for a given number of workstations. Type 3-5 corresponds to maximization of workload smoothness, maximization of work relatedness and multiple objectives, with Type 3 and 4 respectively (Kim *et al*., 1996). Type-E is the most general problem version, which tries to maximize the line efficiency by simultaneously minimizing the cycle time and a number of workstations. While the last, Type-F is an objective-independent problem, which is to establish whether or not a feasible line balance exists for a given combination of m and c.

Furthermore, several version of ALB problem also arise based on the problem structure. It can be classified into two groups. The first group are (Becker and Scholl, 2006; Scholl, 1999), divided into Single-Model Assembly Line Balancing (SMALB) which is involve only one product, Multi-Model Assembly Line Balancing (MuMALB) which is involve more than one product produced in batches and Mixed-Model Assembly Line Balancing (MMALB) which is refer to assembly lines, which are not in batches. Meanwhile, the second group (Baybars, 1986) is divided into Simple Assembly Line Balancing (SALB), which is involves only one product, with features such as paced line with fixed cycle time, deterministic independent processing times, no assignment restrictions, serial layout, one sided workstations, equally equipped workstations and fixed rate launching and General Assembly Line Balancing (GALB), which is include cost function, equipment selection, paralleling, U-shape line layout and mixed-model production.

In another survey of assembly line research (Falkenauer, 2005), there is an identification of additional difficulties (with respect to SALBP) that must be tackled in a line balancing tool, in order to be applicable in those industries and it may be become a reason why current researches should evolve towards in formulating and solving Generalized Problems (GALBP) with different additional characteristics such as cost functions, equipment selection, paralleling, U-shaped line layout and mixed-model production. They are: do not balance but re-balance, workstation identities, un-moveable operations and zoning constraints, cannot eliminate workstations, loads equalization, multiple operators, multi-operator operations, ergonomic constraints (operator positions), multiple products and drifting operations. Figure 3 shows several problem versions which arise from varying the objective in SALBP.

From Fig. 3, few facts could be concluded as listed:

- SALBP-E maximizes the line efficiency E
- SALBP-1 minimizes the number m of stations given the cycle time c
- SALBP-2 minimizes cycle time c given number m of solutions
- SALBP-F seeks for a feasible solution given m and c

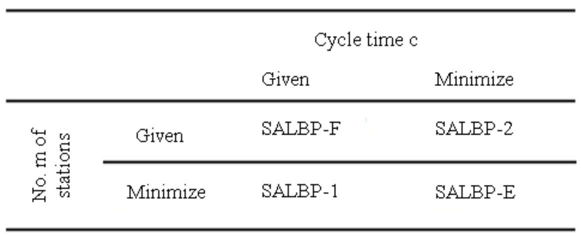
Based on a survey study by Boysen *et al*. (2008), assembly lines balancing problems are classified into a few groups in order to assign typical attributes to different aspects of real-world assembly systems. By doing so, joint occurrences of SALBP-extensions can be identified which are especially characteristic for certain groups of assembly systems in the real world. They are classified based on number models (single, mixed and multi-model), line control (paced line, un-paced synchronous and un-paced asynchronous), frequency (full-time installation and reconfiguration), level of automation (manual lines and automated lines) and lines of business (automobile production). Furthermore, a comparison with the existing literature can clarify if solution procedures for these typical cases already exist or if their development remains for future research. Figure 4 shows a classification of assembly lines balancing problems made by Boysen *et al*. (2008).

On the other hand, due to very different conditions in industrial manufacturing, assembly line production systems and corresponding ALB problems show a great diversity (Boysen *et al*., 2007). For other descriptions of assembly systems and different balancing problems, please refer to, e.g., Buxey *et al*. (1973); Baybars (1986); Shtub and Dhar-El (1989); Ghosh and Gagnon (1989); Erel and Sarin (1998); Scholl (1999) and Rekiek and Delchambre (2001), as well as the most recent survey of Becker and Scholl (2006). Table 2 summaries the classifications of assembly line balancing problem researches up to now. It is an adaptation from Tasan and Tunali (2008) and few modifications are made to complete it, based on the current surveys and investigations.

It is noted that Scholl and Becker (2006) present a survey on problem and methods for GALPB (seems quite similar with Baybars (1986)) with features such as cost/profit-oriented objectives, equipment selection/ process alternatives, parallel workstations/tasks, U-shaped line layout, assignment task processing times and mixed-model assembly lines.

RESULTS

The important research of assembly lines balancing: Assembly lines have been studied extensively (since Salveson, 1955) by introducing his mathematical modeling and looking to the original aim of assembly lines. Balancing assembly lines is a recurring task in operation management, where such models have been used to support the decision maker in configuring an efficient assembly system to optimize productivity (Scholl and Klein, 1999; Boysen *et al*., 2007), which depends on the kinds of assembly lines as classified in Fig. 3, 4 and Table 2, so it became important and carries more benefits and advantages by optimizing assembly lines in order to optimize the productivity.



		Cycle time c	
		Given	Minimize
No. m of stations	Given	SALBP-F	SALBP-2
	Minimize	SALBP-1	SALBP-E

Fig. 3: Version of simple assembly line balancing problem (redrawn Becker and Scholl, 2006)

Table 2: Classifications of assembly line balancing problem

According to survey by Chow (1990)	Based on objective function Scholl (1999)	Based on problem structure Baybars (1986)	Scholl (1999); Becker and Scholl (2006)	Boysen *et al*. (2008)
Multiple products	Type-E	Simple Assembly Line Balancing	Single-Model Assembly Line Balancing	Number models
Variable process time	Type-F	(SALB)	(SMALB)	Line control
Multiple workstation	Type 1 and 2	General Assembly Line Balancing	Multi-Model Assembly Line Balancing	Frequency
Human factors	Type 3-5	(GALB)	(MuMALB)	Level of automation
Product characteristics			Mixed-Model Assembly Line Balancing	Line of business
Length of cycle time			(MMALB)	

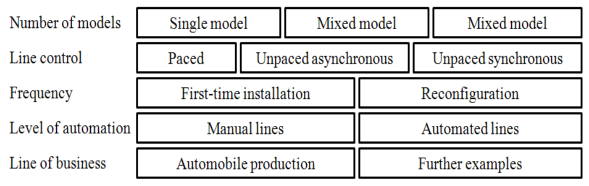


Fig. 4: Kinds of assembly lines (redrawn Boysen *et al*., 2008)

A well-known assembly design problem is the Assembly Line Balancing Problem (ALBP), which deals with the allocation of tasks among workstations so that the precedence relations are not violated and the given objective function is optimized.

ALPB falls into the non-deterministic polynomial hard (NP-hard) class of combinatorial optimization problems (Karp, 1972). The complexity of the ALBP renders optimum seeking methods impractical, for instance, of more than a few tasks and/or workstations, with assumptions there are m tasks and r precedence constraints and then there are $\frac{m!}{2^r}$ possible task sequences (Baybars, 1986). Therefore, it can be time-consuming for optimum seeking methods to obtain an optimal solution within this vast search space. This fact also carries out to a conclusion that researches in assembly lines is very important to do. Even so, many attempts have been made in the literature to solve the ALBP using optimum seeking methods, such as: Linear programming by Salveson (1955), Integer programming by Bowman (1960), Dynamic programming by Held *et al*. (1963) and Branch-and-bound approaches by Jackson (1956). However, none of these methods have been proved to be of practical use for large problems, due to their computational inefficiency (Tasan and Tunali, 2008).

Furthermore, based on surveys by Tasan and Tunali (2008), Genetic Algorithms (GAs) received an increasing attention from researchers, since it provides an alternative to the traditional optimizations technique by using directed random searches to locate optimum solutions in complex landscapes. Few surveys also have been made regarding the subject, namely Dimopoulus and Zalzala (2000) who reviewed the use of evolutionary computation methods for solving manufacturing optimization problems, including the classic job-shop and flow-shop scheduling problems, assembly line balancing and aggregate production planning, Aytug *et al*. (2003) who have reviewed over 110 papers using genetic algorithms to solve various types of production and operations management problems including production planning and control, facility layout design, line balancing, and supply chain management. They noted that none of these studies placed an adequate amount of emphasis on the use of genetic algorithms for solving ALBPs, since their scope was very broad., Scholl and Becker (2006) who presented a review and analysis of exact and heuristic solution procedures for solving ALBPs and Tasan and Tunali (2008) who presented the latest survey on it which conducts to the recent published literature on ALB including genetic algorithms and summarized the main specifications of the problems studied, the genetic algorithm suggested and the objective functions used in evaluating the performance of the genetic algorithms.

However, since there are many efforts by previous researchers in finding the best techniques in solving optimization problems in assembly lines balancing and of course there still wide open problems in ALB, those facts showed that research on assembly lines balancing is important and needs to be addressed in future. Furthermore, it will be important in manufacturing and for sure it is also of paramount importance in the industrial production of high-quantity standardized commodities (Boysen *et al*. 2007).

Soft computing: Soft computing, according to Bonissone (1997) refers to a fusion of techniques that mainly bring together neural networks, fuzzy logic, and evolutionary algorithms (Dubois and Prade, 1998). Here, we present a brief discussion of soft computing and their general capabilities. Furthermore the applications of soft computing in assembly lines balancing are discussed.

An overview of soft computing: The three techniques mentioned above are known as traditional technologies in soft computing. But nowadays, many more novel techniques in soft computing are arisen from behavioral studies, such as ant colony optimization, small world theory, and memes theory (Ovaska *et al*., 2006). Current results have concluded that these technologies have steadily changed the way to solve real-world problems in science and engineering.

The way soft computing techniques are used in solving problems differ to the way that traditional computer algorithms are used. Soft-computing techniques have the ability to generate solutions for many computationally difficult problems (Chaudhari *et al*., 2006). However, in the midst of deployment of soft computing techniques to solve many such problems, it has also given rise to many fundamental questions that are of interest to the discipline of computer science. While many soft computing techniques have attempted to give solutions to specific problems, it is not clear how this approach is generalized for solving all computational problems. Here are four such questions (Chaudhari *et al*., 2006):

- Is the given soft-computing technique general enough, in the sense that, is it possible to express any arbitrary computation in that technique
- Does the given soft-computing technique possess the ability of automatically generating a solution to any arbitrary problem for which an algorithm is known to exist
- Does the given soft-computing technique possess the ability of generating automatically the most efficient solution to an arbitrary computable problem
- When the given soft-computing technique does not generate the most efficient solution, does it generate a reasonably efficient solution, with the performance bound on how far the resulting solution would be from the most efficient solution?

However, few researchers (e.g., Turchin's meta-computations (1993 and 1996a) and super-compilations (Turchin, 1996b), Mitchell's investigations (1994) for cellular automata computations have attempted to answer those questions, even if it has been very difficult. Another interesting field in soft computing is evolutionary computation. Evolutionary computing is based on the concepts of biological evolutionary theory that mimics the mechanics of reproduction, mutation, recombination, natural selection, and survival of the fittest. Three basic kinds of evolutionary computations are genetic algorithms, genetic programming and evolutionary algorithms. Follow we present a brief introduction to the computational capabilities of some soft computing frameworks (adopted from Chaudhari *et al*., 2006).

Turing machine: There is a famous list of nineteenth century problems by Hilbert which is "Does there exists an algorithm for deciding whether or not a specific mathematical assertion does or does not have a proof?" (Weisstein, 1999). Alan Turing, in 1937, showed that the answer for this problem is negative for elementary number theory. In the process of obtaining the solution to this problem, he invented the formalism of "Turing Machine", which is now accepted as (one of the models) to represent any arbitrary computation; in fact, it is an accepted notion today that the problems which can be "computed" are precisely the ones for which a Turing Machine exists (Chaudhari *et al*., 2006). Turing machines are not assigned as a practical computing technology, but a thought experiment about the limits of mechanical computation. Thus, they were not actually constructed. Studying their abstract properties yields many insights into computer science and complexity theory.

Neural networks: A neural network is an artificial system that aims to perform intelligent tasks similar to those performed by the human brain (Pitts and McCullough, 1947). A neural network stores its knowledge through learning within inter-neuron connection strengths known as synaptic weights. These networks have shown themselves to be adept at solving function approximation including time series prediction, fitness approximation and modeling, data processing including filtering, clustering, and also nonlinear controller. The most common neural network model is Multi-Layer Perceptron (MLP). The MLP and other neural network models can be trained using a learning algorithm such as (error) back-propagation, steepest descent, least square error, genetic algorithm, evolutionary computation, expectation-maximization and non-parametric methods. Using one of these algorithms, the weights are determined and the network is said to be trained for a set of data.

Genetic algorithms: Genetic Algorithms (Goldberg, 1989) are based on the Darwinian-type survival of the fittest strategy with sexual reproduction and Mendel's theory of genetics as the basis of biological inheritance. In these theories, stronger individuals in the population have a higher chance of creating offspring. Each individual in the population represents a potential solution to the problem to be solved. Genetic algorithms do not work with a single point on the problem space but use a set, or population of points to conduct a search. This gives genetic algorithms the power to search multi-modal spaces littered with local optimum points.

Genetic algorithms can be used to train a multi-layer perceptron in which weights form a parameter space. While genetic algorithms have the advantage of not getting stuck in local optima, they have other problems. When the search space is very large then genetic algorithm methods generally take a long time to converge to good quality solutions. The length of the search is due to the optimal generalization of the training process with no-prior knowledge about the parameter space.

Evolutionary computing: Evolutionary Computation (EC) has become a standard term to denote a very broad group of algorithms and techniques that are based on the principles of natural processes involving biological evolution. Evolutionary Algorithms (EAs) are mainly meta-heuristic and optimization methods that share some generic concepts borrowed from the natural process of biological evolution. Research in this area has mainly been focused on solving the problems which can be formulated as an exhaustive search over the space of all possible solutions. Using evolutionary computing frameworks, many approaches have been proposed in the last decade. Some approaches for global optimization algorithms include the approaches based on evolution of species (Davis *et al*., 1999), immune system (Castro *et al*., 2002), social behavior of ants (Bonabeau *et al*., 2000), memetic and cultural evolution (Ong *et al*., 2004; Ong *et al*., 2006). Many variants of ECs are also studied by various researchers. For example, Boettecher and Percus (2001) proposed a new optimization algorithm that is based on the principles of natural selection, but it does not follow the basic genetic algorithm framework for population reproduction. Their approach is one step towards integrating different models like principles of self-organized criticality of Bak and Sneppen (1993) in a broad EC framework.

DISCUSSION

Soft computing application: Based on a survey of published theoretical and application literature, it can be concluded that soft computing applications have been used and developed in many research fields and industry. Few of them are automotive and manufacturing, bioinformatics, phylogenetics, computational science, engineering, economics, chemistry, manufacturing, mathematics, physics and other fields, such as neural networks, which are data-driven self-adaptive methods without depending much on prior knowledge about the structural relationship between demand forecasts and the determining factors, can approximate any continuous function arbitrarily well to any given accuracy (Pinkus, 1999). Other than the financial field, a major application of neural networks-based forecasting is in electricity load consumption study (Zhang, *et al*., 1998). As an approximator, similar to neural networks, fuzzy systems can also approximate any continuous function to any degree of accuracy (Ying, *et al*., 1999). Although the performance is similar, neural networks, which are known for their simplicity and model-free approach, have been well accepted in practice and used by many utilities for load forecasting (Hippert *et al*., 2001; Khotanzad, *et al*., 1998) and also there are many other soft computing applications for real-world problems.

Soft computing in assembly lines balancing: It is reported that soft computing approaches have been used (among of them are fuzzy logic and genetic algorithms) in solving assembly lines balancing problems and it is also reported that genetic algorithms have been dominantly used. The uses of genetic algorithms received increasing attention from the researchers, since it provides an alternative to traditional optimization techniques by using directed random searches to locate optimum solutions in complex landscapes (Tasan and Tunali, 2008). In here, we discuss few previous works which used soft computing approaches in solving the assembly line balancing problems, and then the discussion followed by those who used genetic algorithms.

Hui *et al*., (2002) proposed fuzzy operator allocation for balance control of assembly lines in apparel manufacturing. In their work, they demonstrated the use of a fuzzy logic-based system in making balance control decisions. The experimental results show the advantages of the fuzzy logic-based approach over traditional methods, with its ability to reach the target production output more consistently. They also developed a system called FOA for operation allocation, based on a set of fuzzy rules and membership functions obtained through interview sessions with human experts. The performance of the FOA system was compared with that of the supervisors in a men’s shirt factory, using a set of data collected over 30 consecutive working days. It was found to outperform the actual supervisors and extends the literature by increased production efficiency of 30%.

Fonseca *et al*. (2005) proposed a work to model and solve the stochastic assembly line balancing problem with a fuzzy representation of the time variables as a viable alternative method. Two widely-used line balancing methods, the Computer Method for Sequencing Operations for Assembly Lines (COMSOAL) and Ranked Positional Weighting Technique were modified and then transformed to solve the ALBP with fuzzy operating times. The fuzzy heuristics were then automated via Visual Basic. Three test example problems from the available literature were used to successfully validate the constructed fuzzy techniques. Thus, a viable alternative approach to solving the stochastic assembly line balancing problem was developed. The experimental results show that the new fuzzy methods are capable of producing solutions similar to and in some cases better than, those reached by the traditional methods.

Kara *et al*. (2009) proposed a binary fuzzy goal programming model for straight assembly line balancing uses and extends the IP model of Talbot and

Patterson (1984) and a BFGP model for U-shaped assembly line balancing uses and extends the IP model of Urban (1998). Some results and advantages are yielded from the proposed model. They are:

- Allow decision-makers to consider the cycle time and the number of workstation goals as imprecise values
- Minimize the number of workstations and the cycle time at the same time in a fuzzy environment
- It is solved using the Chang's (2007) primary BFGP method
- It is valid and useful for straight and U-shaped assembly line balancing problems
- Enable decision-makers to simultaneously consider conflicting objectives of assembly line balancing in a fuzzy environment
- Allow decision-makers to assign priorities to the goals using weighted goal programming approach
- All these aspects to enable the proposed models to be significant and integrated approaches for assembly line balancing
- The combinatorial nature of the assembly line balancing problems makes the development of fast and effective heuristics significant

Based on the BFGP approaches proposed in this study, the development of heuristics can be considered as topics for further researches.

Genetic algorithm in assembly line balancing: The discussion according to this issue was adopted from the survey by Tasan and Tunali (2008). However, there are also few additional surveys for completing. The survey made is based on the classification given in Baybars (1986) which is to identify the major trends in types of problems studied. Figure 5 represents the structural framework for reviewing, which is done by Tasan and Tunali (2008). However, the discussions only focus on the uses of genetic algorithms for solving ALBP, based on problem specifications only.

Since the research on the ALB problems which used genetic algorithms in solving the problems is much heavier than the other soft computing techniques, so that the discussions are divided into two groups, namely SALBP and GALBP. I will start with the first one, which is research on SALBP problem.

Research on SALB problem regarding the use of genetic algorithms: An assembly line consists of workstations k = 1,…m which are usually arranged along a conveyor belt or similar mechanical material handling equipment. The workpieces (jobs/tasks) are consecutively launched down the line and are moved from station to station. At each station, certain operations are repeatedly performed regarding the cycle time. Cycle time is a maximum or average time available for each work cycle. The basic problem described so far is called a Simple Assembly Line Balancing Problem (SALBP) in the literature (Baybars, 1986). We focus the discussion on SALB problems which used genetic algorithms or a hybrid system on it to solve the problem and continued by the discussion on GALB problem which used genetic algorithm.

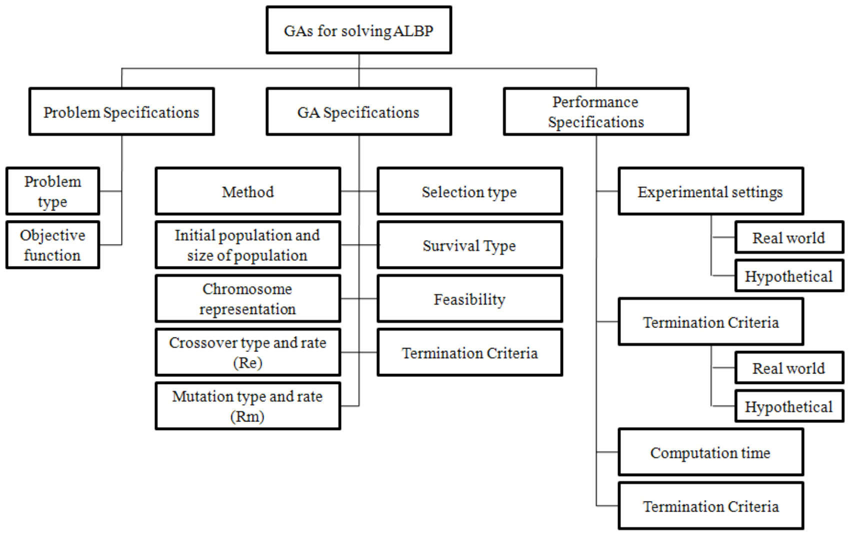


Fig. 5: The structural framework for reviewing GAs in assembly line balancing by Tasan and Tunali (2008) (redrawn)

Falkenauer and Delchamber (1992) were the first to solve the SALB problem with Genetic Algorithms (GAs). Falkenauer (1991) presented the Grouping Genetic Algorithm (GGA) especially for solving grouping optimization problems, where the aim was to group members of a set into a small number of families in order to optimize objective function under given constraints. GGA has a special chromosome representation scheme and genetic operators, which are used to suit the representation scheme. Later, Falkenauer and Delchambre (1992) implemented the GGA to two grouping optimization problems; i.e., a bin packing problem and a SALB Type-1 problem. This study was the first attempt to balance an assembly line Type-1 problem with a genetic algorithm. The authors first presented a special representation scheme and special genetic operators for the bin packing problem and they later modified the special genetic operators for line balancing. Other implementations of GGA for solving ALBPs can be found in Falkenauer (1997); Rekiek *et al*. (1999) and Brown and Sumichrast (2005).

After Falkenauer and Delchambre (1992), the SALB problem was also studied by many researchers. Leu *et al*. (1994) developed a genetic algorithm to solve SALB Type-1 problems and used heuristic procedures to determine the initial population. They also proposed a number of techniques to deal with the feasibility problems during initialization of the population as well as after the reproduction phase. They also demonstrated the possibility of balancing assembly lines with multiple criteria and zoning constraints.

The first article, which presented a genetic algorithm application to the SMALB Type-2 problem, was published by Anderson and Ferris (1994). The authors mainly aimed at showing the effective use of genetic algorithms in solving combinatorial optimization problems. They first described a fairly typical serial implementation of a genetic algorithm for the ALBP and studied the effects of various genetic algorithm variables on the performance of the genetic algorithm. Later, they introduced an alternative parallel version of the genetic algorithm, where each individual in the population resided on a processor. The comparative study between serial and parallel genetic algorithms showed that the quality of the solutions from the parallel implementations was worse than the best solutions obtained from serial implementation.

Rubinovitz and Levitin (1995) used a genetic algorithm to obtain a SALB Type-2 problem, in which the processing time of a task was dependent upon workstation assignment. The authors compared the proposed genetic algorithm to Dar El and Rubinovitz MUST (1979), where the proposed genetic algorithm solved the problems involving more than 20 workstations faster than MUST. Finally, the authors concluded that their genetic algorithm achieved its greater advantage when the precedence constraints were the least restrictive.

Kim *et al*. (1996) developed a genetic algorithm to solve multiple objective SMALB problems. They addressed several types of ALBP, such as to minimize the number of workstations (Type-1), minimize the cycle time (Type-2), maximize workload smoothness (Type-3), maximize work relatedness (interrelated tasks are allotted to the same workstation as much as possible) (Type-4) and a multiple objective with Type 3 and 4 (Type-5). The authors placed the emphasis on seeking a set of diverse Pareto optimal solutions. Although, Kim *et al*.'s (1996) multi-objective genetic algorithm seems to be very promising, the chromosome representation scheme they used is not well suited to the some of the problem types, since they used a single chromosome representation scheme to represent all of the problem types.

Kim *et al*. (1998) considered maximizing the workload smoothness, which has been generally neglected in the literature. Extensive computational experiments were made and the advantages of incorporating problem-specific heuristics information into the algorithm were demonstrated. The experimental results showed that the proposed genetic algorithm outperformed the existing heuristics and the standard genetic algorithm.

Rekiek *et al*. (1999) proposed a grouping genetic algorithm by Falkenauer and Delchambre (1992) based on an Equal Piles approach for solving the SALB problem. They tried to assign tasks to a fixed number of workstations in such a way that the workload of each workstation was nearly equal by leveling on average the size of each workstation (minimizing the standard deviation of sizes). Therefore, the proposed method warranted obtaining the desired number of workstations and tried to equalize the workloads of workstations as much as possible. Later, Rekiek *et al*. (2001) developed a grouping genetic algorithm for solving multi-objective assembly line design problems.

Bautista *et al*. (2000) considered the SALB problem with incompatibilities between tasks. To avoid assigning two incompatible tasks to the same station, the authors developed a Greedy Randomized Adaptive Search Procedure (GRASP), obtained from the application of some classic heuristic methods and a genetic algorithm. They first tried to solve the SALB Type-1 problem and then the SALB Type-2 problem once the number of workstations has been determined.

They also revised GRASP by using weights and called it Greedy Randomize Weighted Adaptive Search Procedure (GRWASP). In the proposed method, the greedy heuristic methods were based on the application of priority rules for assignment of tasks to workstations such as the longest processing time and the greatest number of immediate successors. The greedy heuristic favors tasks with the best index value, while the genetic algorithm phase simply changes the order of elements in the solution. Their comparative study showed that the proposed genetic algorithm and GRWASP resulted in better performance than the greedy heuristics and GRASP.

Ponnambalam *et al*. (2000) developed a multi-objective genetic algorithm for SMALB Type-1 problems to optimize several objectives simultaneously: the number of workstations, the line efficiency and the smoothness index. Several comparisons were made between other heuristics on several examples. The results of the comparisons indicated that the genetic algorithm performed better in all cases studied. However, the execution time for the genetic algorithm was found to be longer.

Sabuncuoglu *et al*. (2000) developed a new genetic algorithm to solve the SMALB problem by utilizing the intrinsic characteristics of the problem. The authors also proposed a method called 'dynamic partitioning' that modified chromosome structure of genetic algorithms to save CPU time. The method modifies the chromosome structure by allocating tasks to workstations (i.e., freezing certain tasks) that satisfy some criteria and continues with the remaining unfrozen tasks. Furthermore, they constructed a new elitism structure adopted from the concept of simulated annealing. It is observed that this new elitism structure contributes significantly to the performance of the genetic algorithm. In fact, the results of extensive computational experiments indicated that the proposed genetic algorithm approach outperformed the well-known heuristics in the literature.

Carnahan *et al*. (2001) considered the physical demands placed on workers in solving the SALB Type-2 problem. In order to measure physical demand, the authors used grip strength capacity that represented the maximum finger flexor strength generated by a worker using a semi-pronated power grip. Three methods, i.e., a ranking heuristic, a combinatorial of the genetic algorithm and a problem space of the genetic algorithm, were developed to simultaneously minimize the maximum manual gripping demands and the cycle time. The authors concluded that the problem space of the genetic algorithm performed better than the others.

Goncalves and De Almedia (2002) presented a hybrid genetic algorithm, which combined heuristic priority rules with a genetic algorithm to solve the SALB Type-1 problem. Several problems from the literature have been used to demonstrate the effectiveness and robustness of the proposed hybrid genetic algorithm. The result of the experiments showed that the proposed method performed remarkably well.

Stockton *et al*. (2004a; 2004b) investigated the use of genetic algorithms for solving various problems that arise when designing and planning manufacturing operations; i.e., assortment planning, aggregate planning, lot sizing within material requirement planning environments, line balancing and facilities layout. In Stockton *et al*. (2004a), the authors have examined the application of a genetic algorithm to the SMALB Type-1 problem. They compared the performance of the genetic algorithm with a traditional solution method, i.e., Ranked Positional Weight (RPW) (Helgerson and Birnie, 1961). In Stockton *et al*. (2004b), the authors performed computational experiments in order to identify suitable genetic operators and parameter values.

Brown and Sumichrast (2005) compared the performance of grouping genetic algorithm GGA by Falkenauer (1991) against the performance of a typical genetic algorithm across a range of grouping problems, i.e., bin packing, machine part cell formation and SALB Type-1 problems. They applied the two techniques, i.e., standard GA and GGA, to a set of problems and compared the results, with respect to solution quality and computation time. They noted that both of the techniques managed to find the optimal solution for all the test problems, however GGA found the optimal solution more quickly.

Research on GALB problems regarding the uses of genetic algorithms: Simply, the discussion on GALB problems is all of the problems that are not SALB. Such as: balancing of single-model or mixed-model, parallel, U-shaped and two-sided lines, with stochastic, fuzzy or dependent processing times.

Tsujimura *et al*. (1995) were the first to solve GALB problems with genetic algorithms. The authors used the fuzzy numbers to represent the imprecise, vague and uncertain task processing times, as the processing times are uncertain due to both machine and human factors. They proposed a genetic algorithm to solve SMALB Type-1 problems, represented the fuzzy processing times by triangular membership functions and illustrated the application of the proposed genetic algorithm on a problem with 80 tasks.

Following Tsujimura *et al*. (1995), several versions of GALB problems were studied by many researchers. Suresh *et al*. (1996) used a genetic algorithm to solve the SMALB Type-1 problem with stochastic processing times. A modified genetic algorithm, working with two populations (one allowing infeasible solutions) and exchange of specimens at regular intervals, were proposed for handling irregular search space (i.e., the infeasibility problem due to problem specifications). The authors believed that a population of feasible solutions would lead to a fragmented search space, thus increasing the probability of getting trapped in a local minimum. They stated that infeasible solutions can be allowed in the population only if the genetic operators can lead to feasible solutions from unfeasible ones. Throughout the generations, some solutions were exchanged at regular intervals between the two populations (i.e., the exchanged solutions have the same rank of fitness value in their own populations). The results of the experiments indicated that the genetic algorithm working with two populations can give better results than the genetic algorithm with only feasible population.

Falkenauer (1997) presented a genetic algorithm based on a Grouping Genetic Algorithm (GGA) by Falkenauer and Delchambre (1992) and a branch-and-bound algorithm for a SMALB Type-1 problem with resource-dependent processing times. The problem involved allocating resources with different cost and speed to each task and also assigning the tasks to workstations, in such a way that the total cost of the line is minimal. The author employed GGA to assign the tasks to workstations and then branch and bound algorithm to select the optimal source for each workstation. In this problem, the processing time of a task depends on the resources being used; therefore, resources with different costs and speeds are allocated to each task in addition to the assignment of tasks to workstations, in such a way that the total cost of the line is minimal. In the proposed method, the tasks were assigned to workstations by GGA and the optimal source for each workstation was selected by a branch-and-bound algorithm.

Ajenblit and Wainwright (1998) were pioneers in balancing the U-shaped SMALB Type-1 problem using genetic algorithms. The authors dealt with two possible variations of this problem; minimizing the total idle time and balancing the workload between workstations, or a combination of both. They developed six different assignment algorithms to interpret a chromosome and assign tasks to workstations. The authors applied the proposed genetic algorithm to 61 test problems. In comparison to previous researchers, they obtained superior results in 11 cases, the same results in 49 cases and a worse result in one case.

Chan *et al*. (1998) proposed a genetic algorithm for a SMALB Type-1 problem in the clothing industry. The authors tried to improve the line efficiency by minimizing the time spent in assembly line balance planning. They also included the various skill levels of workers as problem-specific information to solve a 41-task ALBP. The experimental results showed that the performance of a genetic algorithm was much better than the performance of the greedy algorithm, which performed optimization by proceeding to a series of alternatives and assigned the most skillful worker to each task.

Kim *et al*. (2000) developed a genetic algorithm for balancing a two-sided SMALB Type-1 problem with positional constraints. Two-sided assembly lines consist of two connected serial lines in parallel, where some task can be performed at one of the two sides of the line, while the others can be performed at either side of the line. In the two-sided assembly lines, the tasks were classified into three types: L (left); R (right) and E (either) type tasks. L-type tasks are easily performed at the left-hand side of the line, similarly R-type tasks are easily performed at the right-hand side of the line and E-type tasks are easily performed at both sides of the line. The performance of the proposed genetic algorithm was compared to integer programming and other heuristic methods by Kim *et al*. (1998b), using five test problems. The results indicated that the proposed genetic algorithm showed a better performance than the heuristics studied. The authors stated that the proposed genetic algorithm can be directly applied to the different versions of the ALBP.

Simaria and Vilarinho (2001a) proposed an iterative search procedure, including a genetic algorithm for a MMALB Type-2 problem with parallel workstations. The proposed genetic algorithm procedure was originally based on the model developed in Simaria and Vilarinho (2001b) for a SMALB Type-2 problem, where the simulated annealing was used as a solution method. The iterative procedure starts with a lower bound of cycle time and successively solves the MMALB Type-1 problem by increasing cycle times. Once a feasible solution is found, the procedure employs a genetic algorithm to decrease the cycle time. Besides minimizing the cycle time, the procedure minimizes the workload balances. The iterative procedure was illustrated using a simple example with two assembly models and 25 tasks.

Chen *et al*. (2002) presented a genetic algorithm approach for assembly planning involving various objectives, such as minimizing cycle time, maximizing workload smoothness, minimizing the frequency of tool change, minimizing the number of tools and machines used and minimizing the complexity of assembly sequences. They classified the assembly line planning problems into line balancing, tooling and scheduling problems. The proposed method was improved by including heuristic solutions into initial population and developing a self-tuning method to correct infeasible chromosomes. Several examples were employed to illustrate the proposed genetic algorithm. Experimental results indicated that the proposed genetic algorithm efficiently yields many alternative assembly plans to support the design and operation of an assembly system.

Miltenburg (2002) solved the assembly line balancing Type-1 problem and sequencing problems simultaneously for mixed model U-shaped assembly lines. They proposed a genetic algorithm to solve the balancing and sequencing problems jointly. The proposed genetic algorithm was found to offer good solutions.

Valente *et al.* (2002) proposed a genetic algorithm to solve assembly line balancing Type-2 problem in a real-world application, a two-sided car assembly line. The solution to the problem involved satisfying the constraints that the length of each workstation was constant. The proposed genetic algorithm was found to reduce the total assembly time of the current line by 28.5%.

Brudaru and Valmar (2004) proposed a hybrid genetic algorithm for solving a SMALB Type-1 Problem. They considered the processing times of tasks as fuzzy numbers like Tsujimura *et al.* (1995). Their hybrid method combined the branch-and-bound and genetic algorithm. The authors presented a special chromosome representation scheme, embryonic representation, which used subsets of solutions rather than the individual solutions. They also proposed a new type of genetic operator called a growing operator to be used for the hybrid genetic algorithm. The proposed hybrid genetic algorithm was found to take a longer computation time, with respect to solution quality.

Martinez and Duff (2004) addressed the U-shaped SMALB Type-1 problem. They first solved this problem using 10 heuristic rules adapted from the simple line balancing problem, such as maximum ranked positional weight, maximum total number of follower tasks or precedence tasks and maximum processing time and compared these heuristic solutions with the optimal solutions obtained from previous researches. Following on, they modified Ponnambalam *et al.* genetic algorithm (2000) and inserted the solutions obtained using these heuristic rules to the initial population. They illustrated the proposed genetic algorithm using Jackson's problem (1956). The results showed that the addition of a genetic algorithm can improve the current solution.

Simaria and Vilarinho (2004) expanded the application of their previous work in Simaria and Vilarinho (2001), where they proposed an iterative genetic algorithm-based search procedure for a MMALB Type-2 problem with parallel workstations. The authors have also conducted a set of computational experiments on a set of generated ALBPs.

Levitin *et al.*, (2006) proposed a genetic algorithm for solving a special kind of SMALB Type-2 problem, i.e., Robotic Assembly Line Balancing (RALB) problem. The authors defined a robotic assembly line, where robots with different capabilities and specializations were assigned to the assembly tasks. Various procedures for adapting the genetic algorithm to the RALB problem, such as a local optimization (hill climbing) work-piece exchange procedure, were introduced. Tests were conducted on a set of randomly-generated problems to determine the most effective genetic algorithm procedure, based on the best combination of parameters.

Table 3: Summary of the previous research

References	Problem type	Tools
Falkenauer and Delchamber (1992)	SALB Type-1	Grouping genetic algorithm
Leu *et al*. (1994)	SALB Type-1	Genetic algorithm with heuristic procedures
Anderson and Ferris (1994)	SMALB Type-2	Genetic algorithm
Rubinovitz and Levitin (1995)	SALB Type-2	Genetic algorithm
Tsujimura *et al*. (1995)	GALB (SMALB Type-1)	Genetic algorithms
Kim *et al*. (1996)	SMALB Type-1, 2, 3, 4, 5	Genetic algorithm
Suresh *et al*. (1996)	GALB (SMALB Type-1)	Genetic algorithm with stochastic processing times
Falkenauer (1997)	GALB (SMALB Type-1)	Grouping genetic algorithm
Ajenblit and Wainwright (1998)	GALB (U-shape SMALB Type-1)	Genetic algorithms
Chan *et al*. (1998)	GALB (SMALB Type-1)	Genetic algorithm
Kim *et al*. (1998a)	SALB Type-2	Genetic algorithm
Rekiek *et al*. (1999)	SALB equal piles	Grouping genetic algorithm based on Equal Piles approach
Rekiek *et al*.(2001)	Multi-objective ALB	Grouping genetic algorithm
Bautista *et al*. (2000)	SALB Type-1, Type-2	GRASP with genetic algorithm and GRWASP with genetic algorithm

Table 3: Continued

Kim *et al*. (2000)	GALB (SMALB Type-1)	Genetic algorithm
Ponnambalam *et al*. (2000)	SMALB Type-1	Multi-objective genetic algorithm
Sabuncuoglu *et al*. (2000)	SMALB	Genetic algorithm with dynamic partitioning
Carnahan *et al*. (2001)	SALB Type-2	Ranking heuristic Combinatorial of genetic algorithm Problem space of genetic algorithm (better than the others)
Simaria and Vilarinho (2001a; 2001b)	GALB (MMALB Type-2) SMALB Type-2	Genetic algorithm Simulated annealing
Chen *et al*. (2002)	GALB (assembly planning Type-2)	Genetic algorithm
Goncalves and De Almeida (2002)	SALB Type-1	Hybrid genetic algorithm (combination of heuristic priority rules with genetic algorithm)
Miltenburg (2002)	GALB (MMALB and sequencing simultaneously Type-1)	Genetic algorithm
Valente *et al*. (2002)	GALB (SMALB Type-2)	Genetic algorithm
Hui *et al*. (2002)	SALBP	Fuzzy logic-based system
Zha and Lim (2002)	Intelligent design and planning of manual assembly workstation	Neuro-fuzzy
Brudaru and Valmar (2004)	GALB (SMALB Type-1)	Hybrid genetic algorithm (combined branch and bound with genetic algorithm)
Martinez and Duff (2004)	GALB (U-shape SMALB Type-1)	10 heuristic rules with genetic algorithm
Simaria and Vilarinho (2004)	GALB (MMALB Type-2)	Iterative genetic algorithm based search procedure
Stockton *et al*. (2004)	SALB Type-1 SMALB Type-1	Genetic algorithm Genetic algorithm with computational experiments
Brown and Sumichrast (2005)	SALB Type-1	Genetic algorithm Grouping genetic algorithm (better than the others)
Fonseca *et al*. (2005)	Stochastic SALBP	Fuzzy representation of the time variables as viable alternative method
Levitin *et al*. (2006)	GALB (SMALB Type-2 for RALB)	Genetic algorithm
Noorul Haq *et al*. (2006)	GALB (MMALB Type-1)	Hybrid genetic algorithm (incorporated the solution from the modified RPW method into genetic algorithm)
Kara *et al*. (2009)	Straight and U-shape ALBP	Binary fuzzy goal programming model

Noorul Haq *et al*., (2006) proposed a hybrid genetic algorithm for solving MMALB Type-1 problems. They incorporated the solution from the Modified RPW (MRPW) method into the genetic algorithms randomly-generated initial population to reduce the search space within the global search space. It was noted that this integration reduced the search time. The authors illustrated the implementation of a hybrid genetic algorithm approach on seven problems and compared the results with the MRPW and the standard genetic algorithm. The results showed that the proposed approach performed better than the standard genetic algorithm. The following Table 3 presents the summaries of the previous work regarding assembly line balancing and the uses of genetic algorithms in solving the problems.

CONCLUSION

Assembly line balancing involves numerous problems such as costs, quality, environmental impact, safety, workers, products, reliability, accuracy, robustness of the system and others. In order to make the system stable and balanced, a good manager should take care of all the factors influencing assembly line balancing, since the first-time installation or when it is needed to reconfiguration them. As an approach, soft computing differs from a traditional method, such a conventional or hard computing, in that soft computing approaches are tolerant of imprecision, uncertainty, partial truth and approximation. In effect, the role model for soft computing is the human mind (Zadeh). With all the capabilities of soft computing, especially in optimizing and the simply use of it, make soft computing suitable for assembly line balancing problems that always attempt to optimize the system. The capabilities of soft computing in optimizing have been proven and it is better than hard computing. On the other hand, soft computing approaches make the optimization process simple to do than using the conventional one, which is impractical and computationally inefficient.

However, the fact is that soft computing has been used by many researches and among the approaches, fuzzy logic and genetic algorithms are already being used in solving assembly line balancing problems and genetic algorithms have become the most used method in solving assembly lines balancing problems. The use of genetic algorithms received increasing attention from the researchers, since it provides an alternative to the traditional optimization technique by using directed random searches to locate optimum solutions in complex landscapes (Tasan and Tunali, 2008).

In Table 3, it could be listed in the findings survey study, based on problem specifications. They could be listed as follows (Tasan and Tunali (2008) with few modifications):

- Almost half of the papers surveyed focused on SALB, the simplest version of assembly line balancing problems, while others half-focused on GALB
- Only four articles surveyed dealt with mixed-model assembly line balancing. They are Simaria and Vilarinho (2001a); Simaria and Vilarinho (2004); Miltenburg (2002) and Noorul Haq *et al*. (2006)
- One of the articles (Miltenburg, 2002) tried to solve balancing and sequencing problems of mixed-model assembly lines simultaneously
- Few of the surveyed papers studied on Type-1 problems which minimized the number of workstations and a few others studied on Type-2 problems which minimized the cycle time, but four of them (Kim *et al*., 1996; Bautista *et al*., 2000; Rekiek *et al*., 2002; Ponammbalam *et al*., 2000) considered the multi object
- Only two articles by Suresh *et al*. (1996) and Fonseca *et al*. (2005) dealt with stochastic; another three, by Tsujimura *et al*. (1995); Brudaru and Valmar (2004) and Kara *et al*. (2009) dealt with fuzzy and all the others dealt with deterministic processing times
- Only one article, by Rubinovitz and Levitin (1995), dealt with workstation-dependent and another one by Falkenauer (1997) dealt with resource-dependent deterministic processing times
- Only one article, by Bautista *et al*. (2000), considered the incompatibilities between tasks
- Only one article, by Carnahan *et al*. (2001), considered the physical demands placed on workers during assembly line balancing
- Only one article, by Levitin *et al*. (2006), considered RALB problems, where robots have different capabilities and specializations
- Only one article, by Hui *et al*. (2002), considered the ability of the assembly line to reach target production output more consistently by a proposed fuzzy logic operator allocation-based approach
- Only one article, by Kara *et al*. (2009), considered straight assembly line balancing and a U-shaped model using binary fuzzy goal programming

However, it is noted that most of the researchers focused on SALB, the simplest version of the problem with a single objective and ignore the recent trends, i.e., mixed-model production and U-shaped lines in the complex manufacturing environments, where ALBP are multi-objective in nature. So, it is clearly known and seen that most of the previous researches of assembly lines balancing did not take into account the human factor. It’s obvious that human factors influence the balancing of assembly lines, since there are still many jobs that prefer to be assigned to human beings, although automated production systems are most widely used. Akagi *et al*. (1983) were the first research to pay attention to it. They proposed a method called the Parallel Assignment Method (PAM) which is an alternative way to increase the production rate (hence lowering the cycle time) by assigning multiple workers to one workstation. The experimental results showed that practical problems which cannot be solved by serial line balancing methods are provided and solved by explaining the effectiveness of PAM and could be use to achieve a higher production rate. However, since then, there is very few researchers have achieved the same results or have developed Akagi *et al*.’s (1983) work further. Another finding also been made during this survey study. There are a many reasons which make genetic algorithms become one of the promising optimization techniques in solving assembly lines balancing problems even better than others in some cases. A few of them are listed in Table 4.

Closing/Recommendation: This study has presented a survey study of assembly lines balancing, the problem classifications and their characteristics. A review on the uses of soft computing approaches in assembly line balancing is presented too, as the main concern of this study. This study shows the great effort made by many researchers to prove the capability of soft computing approaches in solving the line balancing problem, rather than using traditional methods such as mathematical modeling and other heuristic methods. This study also shows the importance of researching assembly line balancing. Regarding the review of previous works on assembly line balancing, this study shows that among the soft computing approaches, GAs have been used predominantly in solving assembly line balancing problems, especially the simplest ones. However, in contrast to the high suitability of genetic algorithms in assembly line balancing for multi-objective problems, some researchers (such as Kim *et al*., 1996; Ponnambalam *et al*., 2000) have proved that GA’s computation time is considerably longer. On the other hand, the multi-objective problems of assembly line balancing are the most current issues that need to be addressed.

Table 4: List of other findings regard the uses of genetic algorithm in assembly line balancing

Researchers	Facts about genetic algorithms
Rubinovitz and Levitin (1995)	A few previous researches have produced several good methods and algorithms for solving assembly lines balancing problems, but most of the methods and algorithms suggested just one solution for assembly lines balancing problems (Talbot, 1986). However, in reality, assembly line design needs to investigate alternative solutions, where preference for work allocation to stations is considered, or constraints other then technological precedence are taken into account. Therefore, genetic algorithms are used, since it has the ability to generate multiple solutions to assembly lines balancing problems. The ability of GAs have been compared to the one of optimization techniques, namely MUST (multiple solutions technique (Dar-El and Rubinovitz, 1979)) which can also generate multiple solutions to assembly lines balancing problems, as Gas do. The results show that GAs are faster than MUST algorithms in generating solutions, even for assembly lines balancing problems with large number of stations and a high flexibility ratio. Genetic algorithm-based assembly line balancing algorithms allows for balancing a line where task times are station-independent. The main characteristic of genetic algorithms which robustness implies high independence between the search process and the problem complexity or size. The procedures solution quality evaluation may be easily changed or modified, providing a desirable flexibility to consider and elements and factors of real assembly line design and balancing.
Kim *et al*. (1996)	A genetic algorithms representation suitable to a wide variety of ALB problems, including multiple objective cases. An efficient decoding method for individual representation of sequence alternatives. A simple and effective repair method to preserve the solution's feasibility. The combinations of genetic operators for various single objectives and In the case of multiple objectives, a selection scheme to produce diverse non-dominated solutions.
Ajenblit and Wainwright (1998)	Genetic algorithms provide the ability to find one or more optimum sequences among $\frac{m!}{2^r}$ possible task sequences with m tasks and r ordering constraints than there are, while it is nearly impossible to obtain an efficient solution using a deterministic algorithm.
Rekiek *et al*. (1999)	The classical assembly lines balancing approach tends to group operations under precedence and cycle time constraints. This generally does not yield to a desired number of balanced stations. As no efficient computational methods leading to the exact solution are known for the proposed problem, generally a heuristic method, namely a Grouping Genetic Algorithm (GGA) is used to tackle it.
Kim *et al*. (2000)	A genetic algorithm is a proper strategy for solving the two ALB problems. Not only does GA find good quality solutions quickly to such complex problems, but it is able to readily deal with constraints imposed on by the features of two-sided lines. Therefore, a new GA, a genetic encoding and decoding scheme and genetic operators suitable for the problem are devised.
Sabungcuoglu *et al*. (2000)	The common characteristic of all the heuristic search methodologies is the use of problem-specific knowledge intelligently to reduce the search efforts. In this context, GAs are intelligent random search mechanisms that are applied to various combinatorial optimization problems, such as scheduling, TSP and ALB. GA can be used as a very effective search technique in solving difficult problems because of its ability to move from one solution set to another and its flexibility to incorporate the problem-specific characteristics. GAs are adaptive methods which can be used to solve optimization problems. In general, the power of GAs comes from the fact that the technique is robust and can deal with a wide range of problem areas. Although GAs are not guaranteed to find the optimal solution, they generally find good solutions within reasonable computational requirements. The effective use of GAs in the solution of combinatorial optimization problems, working specifically on the ALB problem. The ability of GAs to consider a variety of objective functions is regarded as the major feature of GAs. Some of the characteristics of GA devise with the inspiration of the ALB system. Coding: Each task is represented by a number that is placed on a string (i.e., chromosome) with the string size equal to the number of tasks. The tasks are ordered on the chromosome, relative to their order of processing. Then the tasks are allocated into stations, such that the sum of the task times in each station does not exceed the cycle time. Fitness function: The objective of the ALB problem considered. Initial population: The initial population is generated randomly by assuring feasibility of precedence relations. Crossover and mutation: The major reason that makes this crossover operator very suitable for ALB is that it assures feasibility of the offspring. Since both parents are feasible, both children must also be feasible. Keeping a feasible population is a key to the ALB problem, since preserving feasibility drastically reduces computational effort. The mutation operator of Leu *et al.* (1994) is scramble mutation; that is, a random cut-point is selected and the genes after the cut-point are randomly replaced (scrambled), assuring feasibility. Elitism, i.e., replacing a parent with an offspring only if the offspring is better than the parent, is applied to both the crossover and the mutation procedures. Both of these operators are the same as Leu *et al*.'s (1994) crossover and mutation operators. Scaling: The objective is to minimize the fitness scores, then it needs to assign the highest scaled fitness score to the lowest fitness score and vice versa, to assign a probability of selection that is proportional to the fitness of chromosomes. Selection Procedure: Each chromosome, consisting of an interval proportional to its scaled fitness score, are placed next to each other on the [0,1] interval. Then, a uniform random number in the [0,1] interval is generated and the chromosome which is assigned to the interval corresponding to the random number is selected. This procedure selects chromosomes proportional to their fitness scores. Stopping Condition: The algorithm terminates after a certain number of iterations.

Finally, this study also provides information to researchers about the problems in assembly line balancing which have been solved and also the ones that are still in progress. This study recommends the following for future research: The human involvement in assembly line balancing needs to be considered as a problem factor; there are still opportunities to use soft computing approaches that have other advantages compared to genetic algorithms, especially for multi-objective problems; to increase the suitability of soft computing approaches, with the hybrid system being one possibility.

ACKNOWLEDGMENT

The researchers honorably appreciate the Malaysian Ministry of Higher Education (MOHE) for the FRGS grant vote number 78366, Malaysian Ministry of Science, Technology and Innovation (MOSTI) and the Research Management Centre (RMC) of University Technology Malaysia (UTM) for the support in making this project successful. The authors also would like to express their cordial thanks to the reviewers for their valuable comments.

REFERENCES

Ahmadi, R. H. and H. Wurgaft, 1994. Design for synchronized flow manufacturing. Manage. Sci., 40: 1469-1483. DOI: 10.1287/mnsc.40.11.1469

Ajenblit, D.A. and R.L. Wainwright,1998. Applying genetic algorithms to the u-shaped assembly line balancing problem. Proceeding of the 1998 IEEE International Conference on Evolutionary Computation, May 4-9, IEEE Xplore Press, Anchorage, Alaska, USA., pp: 96-101. DOI: 10.1109/ICEC.1998.699329

Akagi, F. H., Osaki and S. Kikuchi, 1983. A method for assembly line balancing with more than one worker in each station. Int. J. Prod. Res., 21: 755-770. DOI: 10.1080/00207548308942409

Anderson and, E. J., M.C. Ferris, 1994. Genetic algorithms for combinatorial optimization: The assembly line balancing problem. ORSA J. Comput., 6: 161-173. DOI: 10.1287/ijoc.6.2.161

Aytug, H. M., Khouja and F.E. Vergara, 2003. Use of genetic algorithms to solve production and operations management problems: A review. Int. J. Prod. Res., 41: 3955-4009. DOI: 10.1080/00207540310001626319

Bak, P. and K. Sneppen, 19993. Punctuated equilibrium and criticality in a simple model of evolution. Phys. Rev. Lett., 71: 4083-4086. DOI: 10.1103/PhysRevLett.71.4083

Bautista, J., R. Suarez, M. Mateo and R. Companys, 2000. Local search heuristics for the assembly line balancing problem with incompatibilities between tasks. Proceedings of the IEEE International Conference on Robotics and Automation, Apr. 24-28, IEEE Xplore Press, San Francisco, pp: 2404-2409. DOI: 10.1109/ROBOT.2000.846387

Baybars, I.,1986. A survey of exact algorithms for the simple assembly line balancing problem. Manage. Sci., 32: 909-932. DOI: 10.1287/mnsc.32.8.909

Becker, C. and A. Scholl, 2006. A survey on problems and methods in generalized assembly line balancing. Eur. J. Operat. Res., 168: 694-715. DOI: 10.1016/j.ejor.2004.07.023

Boettcher, S. and A.G. Persus, 2001. Optimization with extremal dynamics. Phys. Rev. Lett., 86: 5211-5214. DOI: 10.1103/PhysRevLett.86.5211

Bonissone, P.P., 1997. Soft computing: The convergence of emerging reasoning technologies. Soft Comput., 1: 1-6. DOI: 10.1007/s005000050002

Bonabeau, E., M. Dorigou and G. Theraulaz, 2000. Inspiration for optimization from social insect behavior. Nature, 406: 39-42. DOI:10.1038/35017500.

Bowman, E.H., 1960. Assembly line balancing by linear programming. Operat. Res., 8: 385-389. DOI: 10.1287/opre.8.3.385

Brown, E.C. and R.T. Sumichrast, 2005. Evaluating performance advantages of grouping genetic algorithms. Eng. Appli. Artifi. Intel., 18: 1-12. DOI: 10.1016/j.engappai.2004.08.024

Brudaru, O. and B. Valmar, 2004. Genetic algorithm with embryonic chromosomes for assembly line balancing with fuzzy processing times. Proceeding of the 8th International Research/Expert Conference Trends in the Development of Machinery and Associated Technology. Neum, Bosnia and Herzegovina.

Boysen, N., M. Fliedner and A. Scholl, 2007. A classification of assembly line balancing problems. Eur. J. Operat. Res., 183: 674-693. DOI: 10.1016/j.ejor.2006.10.010

Boysen, N., M. Fliedner and A. Scholl, 2008. Assembly line balancing: Which model to use when. Int. J. Prod. Econ., 111: 509-528. DOI: 10.1016/j.ijpe.2007.02.026

Buxey, G.M. N.D. Slack and R. Wild, 1973. Production flow line system design-a review. AIIE Trans., 5: 37-48. DOI: 10.1080/05695557308974880

Carnahan, B.J., B.A. Norman and M.S. Redfern, 2001. Incorporating physical demand criteria into assembly line balancing. IIE Trans., 33: 875-887. DOI: 10.1080/07408170108936880

Castro, L. N. and J. Timmis, 2002. An artificial immune network for multimodal function optimization. Proceedings of the IEEE Congress on Evolutionary Computation (CEC’02), IEEE Press, Hawaii, pp: 669-674.

Chan, C.C.K., P.C.L. Hui, K.W. Yeung and F.S.F. Ng, 1998. Handling the assembly line balancing problem in the clothing industry using a genetic algorithm. Intl. J. Cloth. Sci. Technol., 10: 21-37. DOI: 10.1108/09556229810205240

Chang, C.T., 2007. Binary fuzzy goal programming. Eur. J. Operation. Res., 180: 29-37. DOI: 10.1016/j.ejor.2006.03.030

Chaudhari, N.S., Y.S. Ong and V. Trivedi, 2006. Computational capabilities of soft-computing frameworks: An overview. Proceeding of the 9th International Conference on Control, Automation, Robotics and Vision, Dec. 5-8, IEEE Xplore Press, Singapore, pp: 1-6. DOI: 10.1109/ICARCV.345433

Chong, E.K.P. and S.H. Zak, 2008. An Introduction to Optimization. 3rd Edn., Wiley-Inter Sciences Series in Discrete Mathematics and Optimization. New York, USA., ISBN: 0-471-75800-0, pp: 479.

Chen, R.S., K.Y. Lu and S.C. Yu, 2002. A hybrid genetic algorithm approach on multi-objective of assembly planning problem, Eng. Appli. Artifi. Intel., 15: 447-457. DOI: 10.1016/S0952-1976(02)00073-8

Chow, W.M., 1990. Assembly Line Design: Methodology and Applications. 1st Edn., Marcel Dekker Inc., New York, USA., ISBN: 0-8247-8322-0, pp: 121.

Dar-El, E.M. and Y. Rubinovitz, 1979. MUST-a multiple solutions technique for balancing single model assembly lines. Manage. Sci., 25: 1105-1114. DOI: 10.1287/mnsc.25.11.1105

Davis, L.D., K. De Jong, M.D. Vose and L.D. Whitley, 1999. Evolutionary Algorithms. In: The IMA Volumes in Mathematics and Applications, Davis, L.D., K. De Jong, M.D. Vose and L.D. Whitley (Ed.). Springer Verlag., Berlin, ISBN: 10: 0387988262, pp: 191-206.

Dimopoulos, C. and A.M. Zalzala, 2000. Recent developments in evolutionary computation for manufacturing optimization: Problems, solutions and comparisons. IEEE Trans. Evolut. Comput., 4: 93-113. DOI: 10.1109/4235.850651

Dubois, D. and H. Prade,1998. Soft computing, fuzzy logic and artificial intelligence. Soft computing-a fusion of foundations. Methodol. Appli., 2: 7-11. DOI: 10.1007/s005000050025

Erel, E., S.C. Sarin, 1998. A survey of the assembly line procedures. Prod. Plann. Control, 9: 414-434. DOI: 10.1080/095372898233902

Falkenauer, E. and A. Delchambre, 1992. A genetic algorithm for bin packing and line balancing. Proceedings of the 1992 IEEE International Conference on Robotics and Automation, Nice, France, pp: 1186-1192. DOI: 10.1109/ROBOT.1992.220088

Falkenauer, E., 1991. A genetic algorithm for grouping. Proceedings of the 5th International Symposium on Applied Stochastic Models and Data Analysis, Apr. 23-26, Granada, Spain, World Scientific Publishing Co. Pte. Ltd., Singapore, pp: 198-206.

Falkenauer, E.,1997. A grouping genetic algorithm for line balancing with resource dependent task times. Proceedings of the 4th International Conference on Neural Information Processing, Nov. 24-28, University of Otago, Dunedin, New Zealand, pp: 464-468.

Falkenauer, E., 2005. Line balancing in the real world. Proceedings of the International Conference on Product Lifecycle Management, (PLM'05), 200x Inderscience Enterprises Ltd., pp: 360-370. http://www.optimaldesign.com/Download/OptiLine/FalkenauerPLM05.pdf

Ferrero, J., 2000. Computing in science and engineering. Comput. Sci. Eng., 2: 94-97. DOI: 10.1109/MCSE.2000.10027

Fonseca, D.J., C.L. Guest, M. Elam and C.L. Karr, 2005. A fuzzy logic approach to assembly line balancing. Mathware Soft Comput., 12: 57-74. http://dmle.cindoc.csic.es/pdf/MATHWARE_2005_12_01_05.pdf

Ghosh, S. and R.J. Gagnon, 1989. A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems. Int. J. Prod. Res., 27: 637-670. DOI: 10.1080/00207548908942574

Goldberg, D.E., 1989. Genetic Algorithms in Search. Optimization and Machine Learning. 1st Edn., Addison-Wesley Professional, ISBN: 10: 0201157675, pp: 229.

Gonçalves, J.F., J.R. De Almeida, 2002. A hybrid genetic algorithm for assembly line balancing. J. Heuristics., 8: 629-642. DOI: 10.1023/A:1020377910258

Grabau, M.R. and R.A. Maurer, 1998. Assembly line balancing when scarp impact the bottom line. Prod. Inventory Manage. J., 39: 16-21.

Groover, M.P., 2008. Automation, Production System and Computer-Integrated Manufacturing. 3rd Edn., Prentice Hall International, Inc., Upper Saddle River, New Jersey, ISBN: 0-13-207073-1, pp: 375.

Gunasekaran, A. and P. Cecile, 1998. Implementation of productivity improvement strategies in a small company. Technovation, 18: 311-320. DOI: 10.1016/S0166-4972(98)00005-4

Helgeson, N.B. and D.P. Birnie, 1961. Assembly line balancing using the ranked positional weight technique. J. Ind. Eng., 12: 394-398.

Held, M., R.M. Karp and R. Shareshian, 1963. Assembly line balancing-dynamic programming with precedence constraints. Operat. Res., 11: 442-459. DOI: 10.1287/opre.11.3.442

Hippert, H. S., C.E. Pedreira and R.C. Souza, 2001. Neural networks for short-term load forecasting: A review and evaluation. IEEE. Trans. Power Syst., 16: 44-55. DOI: 10.1109/59.910780

Hui, P.C.L., C.C. Keith, K.W. Chan, Yeung and S.F. Frency, N.G., 2002. Fuzzy operator allocation for balance control of assembly lines in apparel manufacturing. IEEE Trans. N. Eng. Manage., 49: 173-180. DOI: 10.1109/TEM.2002.1010885

Jackson, J.R., 1956. A computing procedure for a line balancing problem. Manage. Sci., 2: 261-272. DOI: 10.1287/mnsc.2.3.261

Karp, R.M., 1972. Reducibility among Combinatorial Problems. In: Complexity of Computer Computations, Miller, R.E. and J.W. Thatcher (Eds.). Plenum Press, New York, pp: 85-133.

Kara, Y., T. Paksoy and C.T. Chang, 2009. Binary fuzzy goal programming approach to single model straight and u-shaped assembly line balancing. Eur. J. Operat. Res., 195: 335-347. DOI: 10.1016/j.ejor.2008.01.003

Khotanzad, A., R.A. Rohani and D. Maratukulam, 1998. ANNSTLF-artificial neural network short-term load forecaster-generation three. IEEE Trans. Power Syst., 13: 1413-1422. DOI: 10.1109/59.736285

Kim, Y.K., Y.J. Kim and Y.H. Kim, 1996. Genetic algorithms for assembly line balancing with various objectives. Comput. Ind. Eng., 30: 397-409. DOI: 10.1016/0360-8352(96)00009-5Kimms, A., 2000. Minimal investment budgets for flow line configuration. IIE Transactions, 32: 287-298. DOI: 10.1080/07408170008963907

Kim, Y. K., Y. Kim and T.O. Lee, 1998. Two-sided assembly line balancing models. Technical report, department of industrial engineering. Chonnam National University, Korea. Cited in. DOI: 10.1016/S0360-8352(01)00029-8

Kim, Y.J., Y.K. Kim and Y. Cho, 1998. A heuristic-based genetic algorithms for workload smoothing in assembly lines. Comput. Operat. Res., 25: 99-111. DOI: 10.1016/S0305-0548(97)00046-4

Kim, Y.K., Y. Kim and Y.J. Kim, 2000. Two-sided assembly line balancing: A genetic algorithm approach. Prod. Plann. Control., 11: 44-53. DOI: 10.1080/095372800232478

Lebefromm, U., 1999. Produktions Management. 4th Edn., Einführung MIT Beispielen Aus SAP R/3., Oldenbourg. München, ISBN: 10: 3486273523, pp: 3.

Leu, Y.Y., L.A. Matheson and L.P. Rees, 1994. Assembly line balancing using genetic algorithms with heuristic generated initial populations and multiple criteria. Dec. Sci., 25: 581-606. DOI: 10.1111/j.1540-5915.1994.tb01861.x

Levitin, G., J. Rubinovitz and B. Shnits, 2006. A genetic algorithm for robotic assembly line balancing. Eur. J. Operat. Res., 168: 811-825. DOI: 10.1016/j.ejor.2004.07.030

Lusa, A., 2008. A survey of the literature on the multiple or parallel assembly line balancing problem. Eur. J. Ind. Eng., 2: 50-72. DOI: 10.1504/EJIE.2008.016329

Martinez, U. and W.S. Duff, 2004. Heuristic approaches to solve the u-shaped line balancing problem augmented by genetic algorithms. Proceedings of the IEEE Systems and Information Engineering Design Symposium, Apr. 16-16, IEEE Xplore Press, Charlottesville, VA., pp: 287-293. DOI: 10.1109/SIEDS.2004.239976

Meyers, F.E. and M.P. Stephens, 2005. Manufacturing Facilities Design and Material Handling. 3rd Edn., Prentice Hall, Pearson Education Upper Saddle River, NJ., ISBN: 0131125354, pp: 106.

Miltenburg, J., 2002. Balancing and sequencing mixed-model u-shaped production lines. Int. J. Flex. Manuf. Syst., 14: 119-151. DOI: 10.1023/A:1014434117888

Mitchell, M., J.P. Crutchfield and P.T. Hraber, 1994. Evolving cellular automata to perform computations: Mechanisms and impediments. Phys. D., 75: 361-391. DOI: 10.1016/0167-2789(94)90293-3

Noorul, H.A., J. Jayaprakash and K. Rengarajan, 2006. A hybrid genetic algorithm approach to mixed-model assembly line balancing. Int. J. Adv. Manuf. Technol., 28: 337-341. DOI: 10.1007/s00170-004-2373-3

Ong, Y.S. and A.J. Keane, 2004. Meta-lamarckian learning in memetic algorithm. IEEE Trans. Evolut. Comput., 8: 99-110. DOI: 10.1109/TEVC.2003.819944

Ong, Y. S., M.H. Lim, N. Zhu and K.W. Wong, 2006. Classification of adaptive memetic algorithms: A comparative study. IEEE Trans. Syst. Man Cybern.-Part B., 36: 141-152. http://ieeexplore.ieee.org/iel5/3477/33385/01580625.pdf

Ovaska, S.J. and B. Sick, 2006. Fusion of Soft Computing and Hard Computing: Applications and Research Opportunities. In: Computational Intelligence: Principles and Practice, Yen, G.Y. and D.B. Fogel (Eds.). IEEE Computational Intelligence Society, USA., ISBN: 0-9787135-0-8, pp: 47-72.

Pinkus, A., 1999. Approximation theory of the MLP model in neural networks. Acta, 8: 143-196. DOI: 10.1017/S0962492900002919

Ponnambalam, S.G., P. Aravindan, G. Naidu and G. Mogileeswar, 2006. Multi-objective genetic algorithm for solving assembly line balancing problem. Int. J. Adv. Manuf. Technol., 16: 341-352. DOI: 10.1007/s001700050166

Pitts, W.H. and W.S. Mc Culloch, 1947. How we know universals: The perception of auditory and visual forms. Bull. Math. Biophys., 9: 127-147. DOI: 10.1007/BF02478291

Rekiek, B., P. de Lit, F. Pellichero, E. Falkenauer and A. Delchambre, 1999. Applying the equal piles problem to balance assembly lines. Proceedings of the 1999 IEEE International Symposium on Assembly and Task Planning, July 21-24, IEEE Xplore Press, Porto, Portugal, pp: 399-404. DOI: 10.1109/ISATP.1999.782991

Rekiek, B., P. de Lit, F. Pellichero, T.L. Eglise and P. Fouda *et al*., 2001. A multiple objective grouping genetic algorithm for assembly line design. J. Intl. Manufactur., 12: 467-485. DOI: 10.1023/A:1012200403940

Rekiek, B. and A. Delchambre, 2001. Assembly line balancing and resource planning: What is done and what is still missing. Proceedings of the 17th International Conference on CAD/ACM and Factory of the Future (CARS and FOF), Durban, South Africa, pp: 86-93.

Rekiek, B., A. Dolgui, A. Delchambre and A. Bratcu, 2002. State of art of optimization methods for assembly line design. Annu. Rev. Control., 26: 163-174. DOI: 10.1016/S1367-5788(02)00027-5

Rubinovitz, J. and G. Levitin, 1995. Genetic algorithm for assembly line balancing. Int. J. Prod. Econ., 41: 343-354. DOI: 10.1016/0925-5273(95)00059-3

Sabuncuoglu, I., E. Erel and M. Tanyer, 2000. Assembly line balancing using genetic algorithms. J. Intel. Manuf., 11: 295-310. DOI: 10.1023/A:1008923410076

Salveson, M.E., 1955. The assembly line balancing problem. J. Ind. Eng., 6: 18-25. DOI: 10.1007/978-1-84800-181-7_7

Simaria, A.S. and P.M. Vilarinho, 2001. A genetic algorithm approach for balancing mixed model assembly lines with parallel workstations. Proceedings of the 6th Annual International Conference on Industrial Engineering Theory, Applications and Practice, (IETAP'01), San Francisco, USA., pp: 1-30.

Simaria, A.S. and P.M. Vilarinho, 2001. The simple assembly line balancing problem with parallel workstations-a simulated annealing approach. Intl. J. Ind. Eng., 8: 230-240.

Simaria, A.S. and P.M. Vilarinho, 2004. A genetic algorithm based approach to mixed model assembly line balancing problem of type II. Comput. Ind. Eng., 47: 391-407. DOI: 10.1016/j.cie.2004.09.001

Scholl, A. and R. Klein, 1999. Balancing assembly lines effectively: A computational comparison. Eur. J. Operat. Res., 114: 50-58. DOI: 10.1016/S0377-2217(98)00173-8

Scholl, A., 1999. Balancing and Sequencing of Assembly Lines. 2nd Edn., Physica-Verlag, Heidelberg, ISBN: 3-79808-1180-7, pp: 20.

Scholl, A. and C. Becker, 2006. State of the art exact and heuristic solution procedures for simple assembly line balancing. Eur. J. Operat. Res., 168: 666-693. DOI: 10.1016/j.ejor.2004.07.022

Shtub, A. and E.M. Dar-El,1989. A methodology for the selection of assembly systems. Int. J. Prod. Res., 27: 175-186. DOI: 10.1080/00207548908942537

Stockton, D.J., L. Quinn and R.A. Khalil, 2004. Use of genetic algorithms in operations management part 1: Applications. Proc. Institut. Mech. Eng.-Part B. J. Eng. Manuf., 218: 315-327. DOI: 10.1243/095440504322984867

Stockton, D.J., L. Quinn and R.A. Khalil, 2004. Use of genetic algorithms in operations management part 2: Results. Proc Institut. Mech. Eng.-Part B. J. Eng. Manuf., 218: 329-343. DOI: 10.1243/095440504322984876

Suresh, G., V.V. Vinod and S. Sahu, 1996. A genetic algorithm for assembly line balancing. Prod. Plann. Control., 7: 38-46. DOI: 10.1080/09537289608930323

Talbot, F.B., J.H. Patterson and W.V. Gehrlein, 1986. A comparative evaluation of heuristic line balancing techniques. Manage. Sci., 32: 430-454. DOI: 10.1287/mnsc.32.4.430

Tasan, S.O. and A. Tunali, 2008. A review on the current of genetic algorithm in assembly line balancing. Int. J. Manuf., 19: 49-60. DOI: 10.1007/s10845-007-0045-5

Tempelmeier, H., 2003. Practical considerations in the optimization of flow production systems. Int. J. Prod. Res., 41: 149-170. DOI: 10.1080/00207540210161641

Tsujimura, Y., M. Gen and E. Kubota, 1995. Solving fuzzy assembly line balancing using genetic algorithms. Comput. Ind. Eng., 29: 543-547. DOI: 10.1016/0360-8352(95)00131-J

Turing, A.M., 1937. Computability and lambda-definability. J. Symbol. Logic, 2: 153-163. http://projecteuclid.org/euclid.jsl/1183383711

Turchin, F.V.,1993. Program transformation with metasystem transitions. J. Funct. Programm., 3: 283-313. DOI: 10.1017/S0956796800000757

Turchin, F.V., 1996. Meta-computation: Meta-system transitions plus super-compilation. Lecture Notes. Comput. Sci., 1110: 481-509. DOI: 10.1007/3-540-61580-6_24

Turchin, F.V., 1996. Super-compilation: Techniques and results. Lecture Notes. Comput. Sci., 1181: 227-248. DOI: 10.1007/3-540-62064-8_20

Valente, S.A., H.S. Lopes and L.V.R. Arruda. Genetic Algorithms for the Assembly Line Balancing Problem: A Real-World Automotive Application. In: Soft Computing in Industry-Recent Applications, Roy, R., M. Köppen, S. Ovaska, T. Fukuhashi and F. Hoffman (Eds.). Springer-Verlag, Berlin, ISBN: 1-85233-539-4, pp: 319-328.

Urban, T.L., 1998. Optimal balancing of u-shaped assembly lines. Manage. Sci., 44: 738-741. DOI: 10.1287/mnsc.44.5.738

Veeramani, R., 2001. Assembly Line Balancing. IE 415, http://ecow.engr.wisc.edu/cgi-bin/get/ie/415/veeramani/courseoutline2001.pdf

Wild, R., 1972. Mass-production Management: The Design and Operation of Production Flow-Line Systems. Jhon Wiley and Sons Ltd, London, ISBN-10: 047194405X, ISBN-13: 978-0471944058 1972, pp: 135.

Weisstein, E.W., Decision Problem. Mathworld-A Wolfram Web Resource. CRC Press and Wolfram Research, Inc. http://mathworld.wolfram.com/DecisionProblem.html

Yaman, R., 2008. An assembly line design and construction for a small manufacturing company. Assembly Automat., 28: 163-172. DOI: 10.1108/01445150810863743

Ying, H.Y., S. Ding, Li and S. Shao, 1999. Comparison of necessary conditions for typical takagi-sugeno and mamdani fuzzy systems as universal approximators. IEEE Trans. Syst. Man Cybernet. Part A., 29: 508-514. http://ieeexplore.ieee.org/iel5/3468/17007/00784177.pdf

Zhang, G., B.E. Patuwo and M.Y. Hu, 1998. Forecasting with artificial neural networks: The state of the artfi. Int. J. Forecast., 14: 35-62. DOI: 10.1016/S0169-2070(97)00044-7