CHAPTER 1

INTRODUCTION

An assembly line is a flow-oriented production system where the productive units performing the operations, referred to as stations, are aligned in a serial manner. The work pieces visit stations successively as they are moved along the line usually by some kind of transportation system, e.g., a conveyor belt. Originally, assembly lines were developed for a cost efficient mass-production of standardized products, designed to exploit a high specialization of labor and the associated learning effects (Shtub and Dar-El, 1989; Scholl, 1999). Since the times of Henry Ford and the famous model-T, however, product requirements and thereby the requirements of production systems have changed dramatically. In order to respond to diversified customer needs, companies have to allow for an individualization of their products. For example, German car manufacturer BMW offers a catalogue of optional features which, theoretically, results in 1032 different models (Meyr, 2004). Multi-purpose machines with automated tool swaps allow for facultative production sequences of varying models at negligible setup costs. This makes efficient flow line systems available for low volume assembly-to- order production (Mather, 1989) and enables modern production strategies like mass-customization (Pine, 1993), which in turn ensures that the thorough planning and implementation of assembly systems will remain of high practical relevance in the foreseeable future. Due to the high level of automation, assembly systems are associated with considerable investment costs. Therefore, the (re)-configuration of an assembly line is of critical importance for implementing a cost efficient production system. Configuration planning generally comprises all tasks and decisions which are related to equipping and aligning the productive units for a given production process, before the actual assembly can start. This includes setting the system capacity (cycle time, number of stations, station equipment) as well as assigning the work content to productive units (task assignment, sequence of operations).

1.1. Assembly Line Balancing (ALB)

SALB problem is the core decision problem in configuration planning in its very basic version. Afterwards, the basic assumptions of SALB are examined of how they have to be adopted for setting the more general assumptions of ALB. That way, a definition of ALB, the field to be classified, can be derived. Among the family of ALB problems, the best known and best-studied is certainly the SALB problem. Although it might be far too constrained to reflect the complexity of real-world line balancing, it nevertheless captures its main aspects and is rightfully regarded as the core problem of ALB. In fact, vast varieties of more general problems are direct SALB extensions or at least require the solution of SALB instances in some form. In any case, it is well suited to explain the basic principles of ALB and introduce its relevant terms. A comprehensive review of SALB and its solution procedures is provided by Scholl and Becker (2006). According to the underlying concept of any SALB formulation, an assembly line consists of k = 1,. . .,m (work) stations arranged along a conveyor belt or a similar mechanical material handling device. The work pieces (jobs) are consecutively launched down the line and are hence moved on from station to station until they reach the end of the line. A certain set of operations is performed repeatedly on any work piece which enters a station, whereby the time span between two entries is referred to as cycle time. In general, the line balancing problem consists of optimally partitioning (balancing) the assembly work among all stations with respect to some objective. For this purpose, the total amount of work necessary to assemble a work piece is split up into a set V = {1, . . .,n} of elementary operations named tasks. Tasks are indivisible units of work and thus each task j is associated with a processing time tj also referred to as task time. Due to technological and/or organizational requirements, tasks cannot be carried out in an arbitrary sequence, but are subject to precedence constraints. The general input parameters of any SALB instance can be conveniently summarized and visualized by a precedence graph. This graph contains a node for each task, node weights which equal the task times and arcs reflecting direct as well as paths reflecting indirect precedence constraints. Fig. 1 shows an example precedence graph with n = 9 tasks having task times between 2 and 9 (time units).

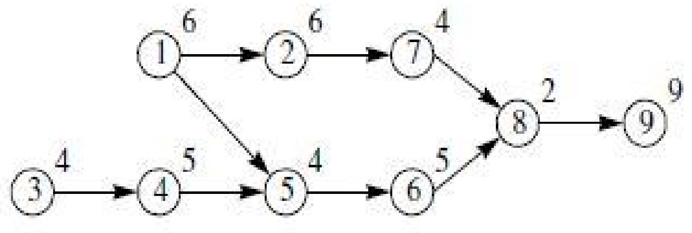

Fig. 1. Precedence graph.

A feasible line balance, i.e., an assignment of tasks to stations, has to ensure that no precedence relationship is violated. The set S_k of tasks assigned to a station k constitutes its station load or work content, the cumulated task time $t(S_k) = \sum_{j \in S_k} t_j$ is called station time. SALB further assumes that the cycle time of all stations is equal to the same value c. Assembly lines with this attribute are called paced, as all stations can begin with their operations at the same point in time and also pass on work pieces at the same rate. As a consequence, all station times of a feasible balance may never exceed c, as otherwise the required operations could not be completed before the work piece leaves the station. Station times can however be smaller than the cycle time, in which case a station k has an unproductive idle time of c- $t(S_k)$ time units in each cycle. For example as shown in Fig. 1, a feasible line balance with cycle time c = 11 and m = 5 stations is given by station loads $S_1 = \{1, 3\}$, $S_2 = \{2, 4\}$, $S_3 = \{5, 6\}$, $S_4 = \{7, 8\}$ and $S_5 = \{9\}$. In order to ensure high productivity, any good balance should cause as few idle times as possible.

1.2 Evolutionary method for optimization

To solve the problem of assembly line balancing, analytical or numerical methods have been applied for computations since a long time to calculate the optimized value. These methods may perform well in many practical cases but they fail in more complex situations. In real manufacturing problems, the number of parameters can be very large and their influence on the value to be optimized (the objective function) can be very complicated having nonlinear character. The objective function may be multimodal (i.e. have many local minimum or maximum), whereas the researcher is always interested in the global optimal values within the search space. Such problems cannot be handled by classical methods (e.g. gradient methods) at all as they converge at local optimal values. In such complex cases, advanced optimization algorithms offer solutions to the problems because they find a solution near to the global optimum within reasonable time and computational effort. These techniques are stochastic in nature with probabilistic transition rules. These techniques are comparatively new and gaining popularity due to certain properties which the deterministic algorithm does not have. The examples include Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Simulated annealing (SA), Artificial Bee Colony (ABC), Teaching Learning Based Optimization (TLBO) etc. A few of them which are used in this thesis are discussed below.

1.2.1 Genetic Algorithm (GA)

GA is an evolutionary algorithm technique which borrows the idea of survival of the fittest amongst an interbreeding population to create a search strategy. It uses only the fitness value and no other knowledge is required for its operation. It is a robust search technique different to traditional algorithms which tend to be more deterministic in nature and get stuck up at local optima. The three basic operators of GA are reproduction, crossover and mutation. Initially, a finite population of feasible solutions to a specified problem is maintained. Through reproduction, it then iteratively creates new populations from the old by ranking the solutions according to their fitness values. Crossover leads to interbreeding the fittest solutions to create new off-springs which are optimistically closer to the optimum solution to the problem at hand. As each generation of solutions is produced, the weaker ones fade away without producing off-springs, while the stronger mate, combining the attributes of both parents, to produce new and perhaps unique off-springs to continue the cycle. Occasionally, mutation is introduced into one of the solution strings to further diversify the population in search for a better solution.

1.2.2 Simulated Annealing (SA)

Simulated annealing is so named because of its analogy to the process of physical annealing of solids in which a crystalline solid is heated and then allowed to cool very slowly until it achieves its most regular possible crystal lattice configuration (i.e. its minimum lattice energy state) and thus is free of crystal defects. If the cooling schedule is sufficiently slow, the final configuration results in a solid with such superior structural integrity. Simulated annealing establishes the connection between this type of thermodynamic behavior and the search for global minima for a discrete optimization problem. Furthermore, it provides an algorithmic means for exploiting such a connection. At each iteration of a simulated annealing algorithm, the objective function generates values for two solutions (the current solution and a newly selected solution) which are then compared. Improved solutions are always accepted while a fraction of non-improving (inferior) solutions are accepted in the hope of escaping local optima in search of global optima. The probability of accepting non-improving solutions depends on a temperature parameter which is typically non-increasing with each iteration of the algorithm. The key algorithmic feature of simulated annealing is that it provides a means to escape local optima by allowing hill-climbing moves (i.e. moves which worsen the objective function value) which occur less frequently as the temperature parameter is finally decreased to zero.

1.2.3 Artificial Bee Colony (ABC)

Inspired by the intelligent foraging behavior of honey bee swarms, the ABC algorithm was introduced to handle unconstrained benchmark optimization functions), similar to other well-known meta-heuristic algorithms. The colony of artificial bees consists of three groups: employed, onlookers, and scout bees. The employed bees randomly search for food-source positions (solutions). Then, by dancing, they share information (communicate) about that food source such as nectar amounts (solutions qualities) with the onlooker bees waiting in the dance area at the hive. The duration of a dance is proportional to the nectar's content (fitness value) of the food source being exploited by the employed bee. Onlooker bees watch various dances before choosing a food-source position according to the probability proportional to the quality of that food source. Consequently, a good food-source position attracts more bees than a bad one. Onlookers and scout bees, once they discover a new food-source position, may change their status to become employed bees. When the food-source position has been visited (tested) fully, the employed bee associated with it abandons it and may once more become a scout or onlooker bee. In a robust search process, exploration and exploitation processes must be carried out simultaneously. In the ABC algorithm, onlookers and employed bees perform the exploration process in the search space while, on the other hand, scouts control the exploration process.

1.2.4 Teaching Learning Based Optimization (TLBO)

The TLBO algorithm is a teaching-learning process inspired algorithm proposed by Rao et al.(2012) and Rao and Patel (2012) based on the effect of influence of a teacher on the output of learners in a class. The algorithm describes two basic modes of the learning: (i) through teacher (known as teacher phase) and (ii) interacting with the other learners (known as learner phase). In this optimization algorithm a group of learners is considered as population and different subjects offered to the learners are considered as different design variables of the optimization problem and a learner's result is analogous to the 'fitness' value of the optimization problem. The best solution in the entire population is considered as the teacher. The design variables are actually the parameters involved in the objective function of the given optimization problem and the best solution is the best value of the objective function. The working of TLBO is divided into two parts, 'Teacher phase' and 'Learner phase'. The TLBO algorithm has been already tested on several constrained and unconstrained benchmark functions and proved better than the other advanced optimization techniques.

All the nature-inspired algorithms such as GA, SA and ABC require algorithm-specific parameters to be set for their proper working in addition to the common control parameters of population size and number of generations. The major advantage with the Teaching Learning based Optimization (TLBO) Algorithm is that it only requires the control over a few common parameters as compared to other evolutionary techniques. This makes the proposed algorithm almost parameter less.

1.3 Research Problem

1.3.1 Objective 1: To identify the factors affecting the assembly line balancing.

Manufacturing domain is classified into many classes, assembly is one of them. The purpose of this dissertation is to find out the factor which affects the efficiency and effectiveness of the assembly line. For this, several research papers from different database have been studied which are shown in chapter 2.

1.3.2 Objective 2: Mathematical modeling for ALBP

Factors identified in objective 1 need to be arranged in manner to solve it for fruitful results. To understand and solve how these factors influences the assembly line balancing, mathematical modeling is required.

1.3.3 Objective 3: ALBP mode optimization using TLBO

This objective uses TLBO as a meta-heuristic technique to solve the mathematical model using the inputs from the research papers available. It shows the iterations of the TLBO and the steps by steps procedure to solve ALBP.

1.3.4 Objective 4: Compare the results with existing problem solution.

Last objective of this dissertation is to compare the results obtained from TLBO to some already existing results. This objective decide whether the new technique i,e TLBO is better than the other or not.

1.4 Thesis overview

The thesis work has been classified into 6 chapters. A brief outline of the chapters is given as:

Chapter 2 contains the review done related to the topic. Review has been done in two steps. Step I collect the research papers on different categories of manufacturing domain and step II is problem specified review of papers related to ALBP.

Chapter 3 describes mathematical modeling of the problem. This section consists of mathematical form of objective defined in the introduction part along with different constraints. It also defines the evaluation criteria and formulae for objective functions.

Chapter 4 explains the optimization technique used in this dissertation to solve the ALBP problem i.e. TLBO. This chapter explain step by steps method of TLBO and come up with different tables which helps to reach some result.

Chapter 5 shows the results obtained from the tables and different runs of TLBO code. This section also compares and discusses the results with the other technique.

Chapter 6 concludes the research work done in this dissertation along with advantages over the other techniques and future research possibilities.

CHAPTER 2

LITERATURE REVIEW

This section of dissertation contains step by step collection and study of research papers available related to the topic. Two databases have been considered namely ScienceDirect and Taylor & Francis to collect the information. Literature review has been done in two steps. Step I tried to identify the problems in manufacturing domain and step II reviews the research paper related to problem identified and techniques used in it. At the end of this section, few objectives for this dissertation work identified and further studies on it done in later chapters.

2.1 Step I: Literature review for problem identification.

2.1.1 Manufacturing Process Planning

- *Artificial Neural Network*

Guh et al. (1999) presents a hybrid intelligent tool (IntelliSPC) in which a neural network based control chart pattern recognition system, an expert system based control chart alarm interpretation system and a quality cost simulation system were integrated for on-line SPC. IntelliSPC was designed to provide the quality practitioners with the status of the process (in control or out-of-control), the plausible causes for the out-of-control situation and cost-effective actions against the out-of-control situation. This tool was intended to be implemented in a scenario where sample data are being collected on-line by automated inspection devices and monitored by control charts. Scheffer et al. (2003) describes an in-depth study on the development of a system for monitoring tool wear in hard turning. Conventional wear-monitoring systems for turning operations cannot be used for monitoring tools used in hard turning because a conglomeration of phenomena, such as chip formation, tool wear and surface finish during hard turning, exhibits unique behavior not found in regular turning operations. various aspects associated with hard turning were investigated with the aim of designing an accurate tool wear monitoring system for hard turning. The findings of the investigation showed that the best method to monitor tool wear during hard turning would be by means of force-based monitoring with an Artificial Intelligence (AI) model. Hsieh. K-H (2010) proposed an integrated procedure incorporating the data mining techniques, e.g. artificial neural networks (ANNs) and stepwise regression techniques, to achieve the construction of yield loss model, the effect analysis of manufacturing process and the clustering analysis of abnormal position (or it can be viewed as defect) for TFT-LCD products.

- *Distributed Artificial Intelligence and fuzzy*

Shih and Srihari (1995) uses a distributed artificial intelligence (DAI) framework to efficiently utilize the infrastructure available for process planning in a batch processing PWB assembly facility. Chu. et al (1996) uses a distributed artificial intelligent (DAI) based system for process planning in surface mount printed circuit board assemble was designed and developed. Multiple intelligent agents work together to read a CAD drawing of an assembly and subassembly deduce the process instruction needed. The system also reviews from a board from a 'Design for Manufacturing' perspective and identify the cost of assembly. It uses fuzzy logic to deal with domain related uncertainty

- *Genetic algorithm and AI planning*

Wong and Chan (2009) describe the development of an effective artificial intelligence technique. Genetic Algorithm (GA), incorporated with an 'earliness' and 'tardiness' production scheduling and planning method to plan the clothing manufacturing process. Additionally, a .segmentation strategy is developed to divide the production-planning period to overcome the problem of chromosome selection in GA. The experimental results demonstrate tbe effectiveness of the proposed method in the clothing manufacturing process. Marchetta and Forradellas (2010) in their paper focus on hybrid procedural and knowledge-based approach based on artificial intelligence planning, which addresses both classic feature interpretation and also feature representation problems

2.1.2 Manufacturing Scheduling

- *Artificial Neural Network*

Vieira and Ribas (2004) provide study of an Artificial Intelligence technique called Simulated Annealing applied to the optimization of production planning problem, more specifically, Master Production Scheduling. This work reviews some of the fundamental theory of simulated annealing, the methodology for master production scheduling calculation, the applicability of simulating annealing to planning problems. Dao. et al (2007) A hybrid Neural Networks have been developed and the simulation techniques have been used to solve complex group scheduling problems. The objectives are to find the best performance criterion optimum for a given number of jobs and machines in order to satisfy different production constraints. The use of combining Hopfield Neural Networks (HNN) and Tabu (local search) approach to define optimal groups of operations which will facilitate the generation of the route sheet. orsoni and Bandinelli (2007) Stochastic events, such as rush orders, stock-out events, and local failures have an important impact on the performance of distributed production, but they are difficult to anticipate and account for when scheduling production activities. Process statistics and artificial intelligence techniques can provide this knowledge to effectively time synchronization events among the simulation and scheduling federates of a same distributed architecture.

- *Artificial Bee colony and swarm intelligence*

Szelke and Markus (1997) presents a combined rule- and case-based reasoning/learning approach to opportunistic reactive scheduling based on a blackboard framework of the system's Expert Supervisor Unit, which also supports human integration into supervisory control of executed processes. Inherent learning ability of the case-based component allows capturing new schedule repair/search control knowledge, including also human preferences, and thus improving the system's reactive/proactive schedule repair efficiency in response to unexpected events/performance deterioration trends during the execution of predictive schedules in manufacturing shop floors. In lee (2001) evaluates artificial intelligence search methods for multi-machine two-stage scheduling problems with due date penalty, inventory, and machining costs and compare four search methods: tabu search, simulated annealing, genetic algorithm, and neighborhood search. Computational results show that the tabu search performs best in terms of solution quality. The tabu search also requires much less computational time than the genetic algorithm and simulated annealing. Banharnsakun. et al (2012) proposes an effective scheduling method based on Best-so-far Artificial Bee Colony (Best-so-far ABC) for solving the JSSP. In this method, the solution direction toward the Best-so-far solution rather a neighboring solution as proposed in the original ABC method. It uses the set theory to describe the mapping of our proposed method to the problem in the combinatorial optimization domain. Madureira et al. (2014) paper presents a novel negotiation mechanism for dynamic scheduling based on social and collective intelligence. Under the proposed negotiation mechanism, agents must interact and collaborate in order to improve the global schedule. Swarm Intelligence (SI) is considered a general aggregation term for several computational techniques, which use ideas and inspiration from the social behaviors of insects and other biological systems. This work is primarily concerned with negotiation, where multiple self-interested agents can reach agreement over the exchange of operations on competitive resources.

2.1.3 Manufacturing System

- *Artificial Neural Network and Fuzzy Logic*

Kullkarni and Kiang (1995) reviews AI-related approaches to Group Technology (GT) and presents the Self-Organizing Map (SOM) network, a special type of neural networks, as an intelligent tool for grouping parts and machines. SOM can learn from complex, multi-dimensional data and transform them into visually decipherable clusters. What sets this technique apart from others in GT is that SOM offers the flexibility of choosing from multiple grouping alternatives. SOM can be used in a dynamic situation where quick response to changes in part designs, process plans, or manufacturing conditions is essential, and thus it can be more easily integrated into a Flexible Manufacturing System. The paper proposes a framework of an intelligent system that integrates the neural networks approach and a knowledge-based system to provide decision supporting functions. Ransing and Lewis (1997) develop a popular Artificial Intelligence tool, `Feedforward Neural Network',. The network is constrained to defect-metacause-rootcause topology and it has been shown that metacause concepts can be successfully associated with the hidden nodes. The errors are calculated at both the output layer and the hidden layer. Although the learning process is based on the back-propagation algorithm with a momentum term, the weight changes would occur at a link connecting a node only if at least one of the nodes connected to it in the preceding layer has non-zero activation. Barschdorff.D et al (1997) describes two hybrid artificial intelligence systems for control and monitoring of manufacturing processes on different hardware and software bases. The first experiences gained by their usage are outlined. Finally, further possible applications of these hybrid solutions in an intelligent manufacturing environment are enumerated. Cheng et al (1998) presents a new approach to implementing agile design and manufacturing concepts. The approach is based on the integration of artificial intelligence (AI) and Internet technologies with the conventional design and manufacturing techniques. Architecture based on AI and Internet programming is proposed for remotely and quickly accessing bearing design and manufacturing expertise at low cost and thus implementing design and manufacturing agility. Chan. et al (2000) proposed an integrated approach for the automatic design of FMS is reported, which uses simulation and multi-criteria decision-making techniques. The design process consists of the construction and testing of alternative designs using simulation methods. The selection of the most suitable design (based on the multi-criteria decision-making technique, the analytic hierarchy process (AHP)) is employed to analyze the output from the FMS simulation models. Intelligent tools (such as expert systems, fuzzy systems and neural networks), are developed for supporting the FMS design process. Active X technique is used for the actual integration of the FMS automatic design process and the intelligent decision support.

- *Adaptive Learning Network (ALN) and AutoCAD*

Kim et al.(2008) examines characteristics of survived small manufacturing enterprises (SMEs) competing to be suppliers to mass merchandisers. It intends to examine various product and management characteristics of small manufacturing enterprises (SME) to determine the critical factors that lead to their long-term survival. Since survey data are usually correlated, fuzzy, inconsistent, and incomplete, they used the adaptive learning network (ALN), an artificial intelligence (AI) technique to build the model. The ALN is non-parametric and known to be much better than multivariate statistical approaches in handling survey data. Kumar And Singh (2008) presents an expert system for automation of strip-layout design process. The proposed system is developed using the production rule-based expert system approach of Artificial Intelligence (AI). It comprises six modules to impart expert advices to the user for identifying sheet metal operations, sequencing of operations, selection of proper piloting scheme, number of stations, staging of operations on progressive die and selection of proper dimensions of stock strip. Finally, the system models the strip-layout automatically in the drawing editor of AutoCAD using the output data files of other modules.

- *Simulated Annealing*

Jozefczyk. J (2006) proposed Heuristic algorithms for solving the task scheduling problem with moving executors to minimize the sum of completion times are considered. The corresponding combinatorial optimization problem is formulated. Three hybrid solution algorithms are introduced. As a basis an evolutionary algorithm is assumed that is combined with the procedure that uses simulated annealing metaheuristics. The results of simulation experiments are given in which the influence of parameters of the solution algorithms as well as of the number of tasks on the quality of scheduling and on the time of computation is investigated.

- *Particle Swarm Optimization*

Ficko at al. (2010) proposed system which is composed of a creative subsystem which can use different evolutionary optimization methods, and a subsystem for evaluating layouts. In the presented work the subsystem for creation uses a particle swarm optimization method for the creation/modification of solution sets. Evaluation of solution quality is made using intelligent search of the shortest travel paths within the layout. This system has proved to be innovative since it proposes very good solutions which are oriented to the main task of the system and are not simplified because of human limitations.

- *Genetic Algorithm and Neural network*

Zeidi et al. (2013) presents a new multi-objective nonlinear programming model in a dynamic environment. Furthermore, a novel hybrid multi-objective approach based on the genetic algorithm and artificial neural network is proposed to solve the presented model. From the computational analyses, the proposed algorithm is found much more efficient than the fast non-dominated sorting genetic algorithm (NSGA-II) in generating Pareto optimal fronts.

2.1.4 Manufacturing Process

- *Artificial Neural Network*

Guessasma. et al. (2004) introduce a new approach based on artificial intelligence responding to these requirements. A detailed procedure is presented considering an artificial neural network (ANN) structure which encodes implicitly the physical phenomena governing the process. The implementation of such a structure was coupled to experimental results of an optic sensor controlling the powder particle fusion state before the coating formation. The optimization steps were discussed and the predicted results were compared to the experimental ones allowing the identification of the control factors. Lorenzo. et al. (2006) proposed artificial intelligence (AI) techniques are applied to ductile fracture prediction in cold forming operations. The main advantage of the application of AI tools and in particular, of artificial neural networks (ANN), is the possibility to obtain a predictive tool with a wide applicability.

- *Fuzzy Logic*

Neuroth. et al. (2000) discusses the application of two AI-based techniques, fuzzy logic and artificial neural networks (ANNs), to specific problems related to the operation of oil and gas transport facilities. Vitanov. et al. (2001) considers the use of combined artificial intelligence and modeling techniques. It includes a new frame of a Neurofuzzy-model based Decision Support System _ *FricExpert*, which is aimed at speeding up the parameter selection process and to assist in obtaining values for cost effective development. Derived models can then be readily used for optimization techniques.

- *Genetic Algorithm and Depth First Search*

Kim and Im (1999) discuss a methodology of applying the searching technique for process sequence design, and the flexibility of the introduced searching technique is evaluated by generating design examples of a shaft part, a wrench and hexagonal bolts of AISI 1045. Wang. et al.(2003) discusses the development and application of a hybrid artificial neural network and genetic algorism methodology to modeling and optimization of electro-discharge machining. The hybridization approach is aimed not only at exploiting the strong capabilities of the two tools, but also at solving manufacturing problems that are not amenable for modeling using traditional methods. Based on an experimental data, the model was tested with satisfactory results. The developed methodology with the model is highly beneficial to manufacturing industries, such as aerospace, automobile and tool making industries.

2.1.5 Assembly

- *Genetic Algorithm*

Leu. et al. (1996) introduced the use of an artificial-intelligence based technique, genetic algorithms (GA), to solve mixed-model assembly-line sequencing problems. It also shows how practitioners can comfortably implement this approach to solve practical problems.

- *Artificial Neural Network*

Altiparmak et al. (2007) developed an artificial neural network (ANN) metamodel for a simulation model of an AAS. The ANN and regression metamodels for each AAS are compared with respect to their deviations from the simulation results. The analysis shows that the ANN metamodels can successfully be used to model of AASs. Consequently, one concludes that practicing engineers involved in assembly system design can potentially benefit from the advantages of the metamodeling approach.

- *Artificial Bee Colony Algorithm*

Tapkan. et al. (2012) proposed two different swarm intelligence based search algorithms are implemented to solve large-sized instances. Bees algorithm and artificial bee colony algorithm have been applied to the fully constrained two-sided assembly line balancing problem so as to minimize the number of workstations and to obtain a balanced line. An extensive computational study has also been performed and the comparative results have been evaluated.

2.1.6 Inventory

- *Particle Swarm Optimization*

Sinha. et al. (2012) uses Immuno- particle swarm optimization with penetrated hyper- mutation (COIPSO-PHM) in inventory replenishment, a new algorithm which uses clonal selection approach in particle swarm optimization by embedding co- evolutionary theory to solve the problem of inventory replenishment in distributed plant–warehouse– retailer system. Constraint handling is explicitly taken care by implanting augmented lagrangian concept.

2.1.7 Material Handling

- *Fuzzy Knowledge- Based System*

Hamid. et al. (2009) presented a hybrid method for the selection and assignment of the most appropriate Material Handling Equipment (MHE). In the first phase, the system selects the most appropriate MHE types for every MH operation in a given application using a Fuzzy Knowledge-Based Expert System consisting of two sets of rules: Crisp Rules and Fuzzy Rules. In the second phase, a Genetic Algorithm (GA) searches throughout the feasible solution space, constituting of all possible combinations of the feasible equipment specified in the previous phase, in order to discover optimum solutions.

2.1.8 Logistic and Maintenance

- *Ant Colony Optimization*

McMullen. P .R (2001) presented an application of the relatively new approach of an ant colony optimization to address a production- scheduling problem when two objectives are present- simulating the artificial intelligence agent of virtual ants to obtain desirable solutions to a manufacturing logistic problem. Two objectives are minimization of setups and optimization of stability of material usage rates.

- *Genetic Algorithm*

Han and Yang (2006) proposed a new e-maintenance system that is dependent upon coordination, co-operation and negotiation through the use of Internet and tether-free (i.e. wireless, web, etc.) communication technologies. This e-maintenance enables manufacturing operations to achieve near-zero downtime performance on a sharable, quick and convenient platform through integrating the existent advanced technologies with distributed sources. The main difference between the proposed e-maintenance and existing systems is the system structure. This e-maintenance consists of two subsystems: maintenance centre and local maintenance. The relationship of both subsystems can be considered as supplier and clients. This division can effectively reduce maintenance cost, maintenance system design period, and solve the problem of lack of experts. Tamayo et al. (2009) tried to solve the problem of minimizing the cost of products. First the raw material dispersion problem is analyzed, in order to determine a risk-level criterion or ‘‘production criticality’’. This criterion is used subsequently to optimize deliveries dispatch with the purpose of minimizing the number of batch recalls in case of crisis. This is achieved by implementing decision-making aid tools based on operational research and artificial intelligence.

- *Artificial Neural Network*

Yang and Lu (2010) addressed a hybrid dynamic pre-emptive and competitive neural-network approach in solving the multi-objective dispatching problem. It optimizes three performance criteria simultaneously, namely: cycle time, slack time, and throughput. To determine appropriate dispatching strategies, under various system conditions, is a non-trivial challenge to control the complex systems.

2.2 Step II: Literature review for mathematical modeling and formulation of problem.

2.2.1 Genetic Algorithm (GA)

Zhao et al. (2015) proposed a mathematical model to formulate the multi-objective mixed-model assembly line (MMAL) problem and the genetic algorithm is applied for problem solving due to the computational complexities. They had used numerical example to demonstrate the effectiveness of the proposed approach. Their results incorporating the impact of mental workload on performance into account can make the rolled throughput yield (RTY) and efficiency balance when designing the MMAL. Moreover, they also verified that improving the experience of the operator scan mitigate the impact of mental workload on the quality and efficiency. In the same year Tiacci. L (2015) proposed an innovative approach to deal with the Mixed Model Assembly Line Balancing Problem (MALBP) with stochastic task times and parallel work stations. At the current stage of research, advances in solving realistic and complex assembly line balancing problem, as the one analyzed, are often limited by the poor capability to effectively evaluate the line throughput. Although algorithms are potentially able to consider many features of realistic problem and to effectively explore the solution space, a lack of precision in their objective function evaluation (which usually includes a performance parameter, as the throughput) limits in fact their capability to find good solutions. The author took a decisive step by coupling the most recent advances of simulation techniques with a genetic algorithm approach. A parametric simulator, developed under the event/object oriented paradigm, has been embedded in a genetic algorithm for the evaluation of the objective function, which contains the simulated throughput. Ramezanian and Ezzatpanah (2015) worked mixed-model assembly line balancing and worker assignment problem (MMALBWAP). Mixed-model assembly lines allow the simultaneous assemble of a set of products on a single assembly line. The worker assignment problem deals with assigning workers to workstations considering their abilities and operating costs. Authors have considered two incoherent objectives. The first objective aims to minimize the total cycle time. From one side different models of product have different operating task times and on the other hand different worker skills make more varieties in operating times, therefore minimizing cycle time in the problems which seems to be important. Simultaneous with cycle time the operating costs related to workers is the second objective of interest to be minimized. To solve this multi-objective problem a goal programming approach is utilized and because of high complexity of the problem, an evolutionary algorithm named imperialist competitive algorithm (ICA) is developed. Alper and Gokalp (2012) proposed a Priority-Based Genetic Algorithm (PGA) based method for the simultaneously tackling of the mixed-model U-shape assembly line (MMUL) for line balancing/model sequencing problems (MMUL/BS) with parallel workstations and zoning constraints. This method allows the decision maker to control the process to create parallel workstations and to work in different scenarios. The method presented simulated annealing based fitness evaluation approach (SABFEA) which is able to make fitness function calculations easily and effectively. A new fitness function is adapted to MMULs for aiming at minimizing the number of workstations as primary goal and smoothing the workload between-within workstations by taking all cycles into consideration. Consecutively Yolmeh and Kianfar (2012) proposed a hybrid Genetic algorithm to solve assembly line balancing and scheduling problem (SUALBSP). The problem deals with assigning the tasks to the stations and scheduled them inside each station. A simple permutation was used by author to determine the sequence of tasks. To determine the assignment of tasks to the stations, the algorithm was hybridized by using a dynamic programming method. Mutlu et al. (2012) considered the assembly line worker assignment and balancing problem of type-II (ALWABP-2). ALWABP-2 arises when task times differ depending on operator skills and concerns with the assignment of tasks and operators to stations in order to minimize the cycle time. Authors developed an iterative genetic algorithm (IGA) to solve this problem. In the IGA, three search approaches are adopted in order to obtain search diversity and efficiency: modified bisection search, genetic algorithm and iterated local search. When designing the IGA, all the parameters such as construction heuristics, genetic operators and local search operators are adapted specifically to the ALWABP-2. B. T. Raghda et al.(2011) developed a Genetic Algorithm (GA) to solve the two-sided assembly line balancing problem. The developed GA specifies a new method for generating the initial population. It applies a hybrid crossover and a modified scramble mutation operators..A proposed station oriented procedure is adopted for assigning tasks to mated-stations. It specifies the side of the either tasks based on proposed side assignment rules rather than assigning them randomly. These rules are effective especially in large problems. The proposed method of generating the initial population is able to generate feasible solution in different areas of the search space. The applied genetic operators are able to preserve the feasibility of all solutions throughout all the developed generations. The proposed GA by the author was capable to find the optimum and near optimum solutions within a limited number of iterations. Akpinar and Bayhan (2011 proposed a hybrid genetic algorithm to solve mixed model assembly line balancing problem of type I (MMALBP-I). There are three objectives to be achieved: to minimize the number of workstations, maximize the workload smoothness between workstations, and maximize the workload smoothness within workstations. The proposed approach was able to address some particular features of the problem such as parallel workstations and zoning constraints. The genetic algorithm may lack the capability of exploring the solution space effectively. The authors improved the exploring capability by sequentially hybridizing the three well known heuristics, Kilbridge & Wester Heuristic, Phase-I of Moodie & Young Method, and Ranked Positional Weight Technique, with genetic algorithm.

Kim et al. (2009) proposed a mathematical model and a genetic algorithm (GA) for two-sided assembly line balancing (two-ALB). The mathematical model was used as a foundation for practical development in the design of two-sided assembly lines. In the GA, they adopted the strategy of localized evolution and steady-state reproduction to promote population diversity and search efficiency. When designing the GA components, including encoding and decoding schemes, procedures of forming the initial population, and genetic operators, they had considered the features specific to two-ALB. Through computational experiments, the performance of the proposed GA wad compared with that of a heuristic and an existing GA with various problem instances. Ana and Pedro (2004) gave a mathematical programming model and an iterative genetic algorithm-based procedure for the mixed-model assembly line balancing problem (MALBP) with parallel workstations, in which the goal is to maximize the production rate of the line for a pre-determined number of operators. The problem taken in his work was basically deals with the operating conditions of real-world assembly lines, like zoning constraints and workload balancing and also allows the decision maker to control the generation of parallel workstations.

2.2.2 Particle Swarm Optimization (PSO)

Yuan et al. (2014) proposed a hybrid Honey bee mating optimization (HBMO) algorithm to solve the mixed-model two-sided assembly line balancing (MTALB) problem with the objective of minimizing the number of mated-stations and total number of stations for a given cycle time. Compared with the conventional HBMO algorithm, the proposed algorithm employs the simulated annealing (SA) algorithm with three different neighborhood structures as workers to improve broods, which could achieve a good balance between intensification and diversification during the search. In addition, a new encoding and decoding scheme, including the adjustment of the final mated-station, is devised to fit the MTALB problem. In the same year Saif et al. (2014) aimed to minimize the cycle time in addition to maximizing the probability that completion time of tasks on stations will not exceed the cycle time and minimize smoothness index simultaneously. A Pareto based artificial bee colony algorithm proposed by the authors to get Pareto solution of the multiple objectives. The proposed algorithm called Pareto based artificial bee colony algorithm (PBABC) introduces some extra steps i.e., sorting of food source, niche technique and preserve some elitists in the standard artificial bee colony algorithm (ABC) to get

Pareto solution. Furthermore, the effective parameters of the proposed algorithm are tuned using Taguchi method. Tapkan et al. (2012) used a mathematical programming model in order to describe the problem formally. Due to the problem complexity, two different swarm intelligence based search algorithms are implemented to solve large-sized instances. Bees algorithm and artificial bee colony algorithm have been applied to the fully constrained two-sided assembly line balancing problem so as to minimize the number of workstations and to obtain a balanced line. Ozbakir and Tapkan (2011) used Bees Algorithm to solve without constrained and zone constrained two-sided assembly line balancing problem with simple neighborhood structures, it was observed that the proposed algorithm performs well, since it finds the best known number of station and yields the other goals effectively. Neighborhood structures are especially determined as simple shift and swap in order to evaluate the performance of Bees Algorithm itself.

2.2.3 Ant Colony Optimization (ACO)

Zha and Yu (2014) worked on U-line rebalancing problem and formalized it with respect to minimization the moving cost of machines and labor cost. The walking time of operators is considered to avoid generating awkward walking path. A new hybrid algorithm of ant colony optimization and filtered beam search are used to solve the problem. The hybrid algorithm adopts the framework of ant colony optimization. In the process of constructing path, each ant explores several nodes for one step and chooses the best one by global and local evaluation at a given probability. Akpinar and Baykasoglu (2014) proposed a mixed-integer linear mathematical programming (MILP) model for mixed-model assembly line balancing problem with setups. In the MILP model, some of the features of the real world problems like parallel workstations, zoning constraints, and sequence dependent setup times between tasks, which is an actual framework in assembly line balancing problems. The main endeavor is to formulate the sequence dependent setup times between tasks in type-I mixed-model assembly line balancing problem. The proposed model considers the setups between the tasks of the same model and the setups because of the model switches in any workstation. Akpinar et al. (2013) proposed a new hybrid algorithm, which execute ant colony optimization in combination with genetic algorithm (ACO-GA), for type-I mixed model assembly line balancing problem (MMALBP-I) with some particular features of real world problems such as parallel workstations, zoning constraints and sequence dependent setup time between tasks. The algorithm aims at enhancing the performance of ant colony optimization by incorporating genetic algorithm as a local search strategy for MMALBP-I with setups. ACO is conducted to provide diversification, while GA is conducted to provide intensification. Sabuncuoglu. I et al. (2009) proposed an ACO approach to solve the single-model U-type assembly line balancing problem (UALBP). The proposed ACO algorithm outperforms the SA and displays very competitive performance against the state-of-the-art ULINO algorithm.

2.2.4 Simulated Annealing (SA)

Jayaswal and Agarwal (2014) worked on the problem of RDULB and proposed a Simulated Annealing (SA) based metaheuristic, which provide optimal solution for most of the small-to-medium problems like to assign not only the task but also resource alternatives (number of workers and equipment type) to workstations which is quite complex as compare to straight assembly line. For very large problems, while SA generates a good feasible solution within half an hour to 1.5 h, Cplex is unable to find a single feasible solution even after 10 times the CPU time required by SA. Hamzadayi and Yildiz (2013) proposed a simulated annealing algorithm for solving a problem of type 1 assembly line balancing problem by ignoring the fixed model sequence. Accordingly, simulated annealing based fitness evaluation approach proposed by authors is enhanced by adding the tabu list, and inserted into the proposed algorithm. Authors have demonstrated the implementation difficulties experienced in meta-heuristics based on solution modification for solving these types of problems. It is found that ADW is an insufficient performance criterion for evaluating the performance of the solutions. Most of the research deals with balancing a mixed-model U-line in a Just-In-Time (JIT) production system. The research intends to reduce the number of stations via balancing the workload and maximizing the weighted efficiency, Manavizadeh et al. (2013) assumed that there are two types of operators: permanent and temporary. Both types can work in regular and overtime periods. Based on their skill levels, workers are classified into four types. The sign at each work station indicates types of workers allowed to work at that station. They have used an alert system using the hybrid Kanban systems. A Simulated Annealing algorithm was applied in the following three stages. First, the balancing problem was solved and the number of stations was determined. Second, workers were assigned to the workstations in which they are qualified to work. Following that, an alert system based on the Kanban system was designed to balance the work in the process inventory. This was achieved by defining control points based on the processing time and making control decisions to minimize the number of kanban cards, In the proposed algorithm, two methods for the temperature cooling schedule were considered and two methods were defined for determining the number of neighborhood search. The initial temperature was considered equal to the cost of the initial solution to reach the convergence situation as soon as possible. Roshani et al. (2013) worked on Assembly line balancing problems with multi-manned workstations usually occur in plants producing high volume products (e.g. automotive industry) in which the size of the product is reasonably large to utilize the multi-manned assembly line configuration. In these kinds of assembly lines, usually there are multi-manned workstations where a group of workers simultaneously performs different operations on the same individual product. Because of the high computational complexity, it is quite difficult to achieve an optimal solution to the balancing problem of multi-manned assembly lines with traditional optimization approaches. So the authors proposed a simulated annealing heuristic algorithm for solving assembly line balancing problems with multi-manned workstations. The line efficiency, line length and the smoothness index are considered as the performance criteria. Khorasanian et al. (2013) worked on two sided assembly line balancing problem and proposed an index for calculating the value of the relationship between each two tasks, and define a performance criterion called 'assembly line tasks consistency' for calculating the average relationship between the tasks assigned to the stations of each solution. They introduced a simulated annealing algorithm for solving the two-sided assembly line balancing problem considering the three performance criteria of number of stations, number of mated-stations, and assembly line tasks consistency. Also, the simulated annealing algorithm is modified for solving the two-sided assembly line balancing problem without considering the relationships between tasks.. Cakir et al. (2011) worked on multi-objective optimization of a single-model stochastic assembly line balancing problem with parallel stations. The authors tried to minimize the smoothness index and design cost. To obtain Pareto-optimal solutions for the problem, they proposed a new solution algorithm, based on simulated annealing (SA), called m_SAA. m_SAA implements a multinomial probability mass function approach, tabu list, repair algorithms and a diversification strategy.

2.2.5 Teaching Learning Based Optimization (TLBO)

Tuncel and Aydin (2014) considered a real life two-sided ALBP with additional assignment restrictions. They took into account operating sides of tasks in addition to precedence and cycle time constraints, when the allocation of the tasks to an ordered sequence of workstations is determined. Moreover, the problem involves several compatible and incompatible zoning constraints. Accordingly, some groups of tasks must be executed together on the same station (compatible tasks) and other tasks were prevented from being assigned to the same station (incompatible tasks). The problem was composed of several tasks which should be assigned to the different stations with no other tasks assigned (negative zoning constraint). Objective function was to minimize number of workstations and to ensure a smooth distribution of workload between workstations. Xu et al. (2015) present first research work of a teaching–learning-based optimization algorithm for solving the flexible job-shop scheduling problem with fuzzy processing time. According to the characteristics of the FJSPF, special encoding and decoding schemes were employed, and special search operators were designed for the teaching, learning and studying phases. Both the computational complexity and the influence of parameter setting were investigated. Numerical results demonstrated the effectiveness of the proposed TLBO by comparing with the existing algorithms in solving benchmark instances.

CHAPTER 3

MATHEMATICAL MODELING FOR ALBP

Among the family of ALBPs, the most famous is certainly the simple assembly line balancing problem (SALBP) which has been studied for several decades. Because of the numerous simplifying assumptions of SALBP, this problem is too constrained to reflect the complexity of practical line balancing (Boysen et al., 2007). Therefore in this thesis, we have used classical SALBP by adding some realistic relevant aspects. This section presents the proposed mathematical model for any common assembly line. Before the notations and the mathematical model are presented, the basic assumptions must be stated in order to completely describe the problem.

3.1 Assumptions

The basic assumptions of the problem are characterized as follows:

1. A homogenous product is assembled by operating n tasks of an assembly line in m workstations.
2. A paced serial line with fixed number of workstations and no buffer is considered.
3. The required time to operate task i is variable between lower bound Lti and upper bound Uti. Note that a multi-objective SALBP under uncertainty of operation time is regarded when there is no prior information about probability distributions of the operation times.
4. Compressing the processing times may lead to higher equipment cost due to cumulative erosion, wear, depreciation and so on. In other words, it may reduce the lifetime of equipment.
5. We study SALBP with learning consideration using learning curve introduced by Biskup (1999), in which the operation time of task i with a learning effect if assigned to position r is defined as:

$$t_{ir} = t_i r^{\alpha} \tag{1}$$

where $\alpha(=\log_2 s \le 0)$ is the learning effect when s is the learning rate.

6. If task k is assigned next to task i at the same workstation, setup sut_{ik} must be added to calculate the global workstation time. The setup times matrix is known deterministically.
7. Operation and setup times are independent on the workstation in which tasks are operated.
8. Since all workstations are equally equipped, we set a fixed value of equipment cost per piece.
9. A task cannot be performed until all its predecessors have been completed.
10. Each task can be assigned to only one workstation
11. Only one task can be processed in each workstation at a time.
12. Tasks must be processed only once.

3.2 Mathematical model

With the mentioned basic assumptions, the mathematical model will be presented using the notations defined in Table 1. The multi-objective SALBP with flexible operation times, sequence-dependent setup times and learning effect can be formulated as the following mixed integer nonlinear programming model.

Table 1: List of notations.

Notation	**Definition**
i, k	Task indices
j	Workstation index
n	Sequence position index inside a workstation
m	The number of workstation
Mn	Maximum number of tasks that can be assigned to any workstation
t_i	Standard operation time of task i
Lt_i	Lower bound of operation time of task i
Ut_i	Upper bound of operation time of task i
t_{ir}	Operation time of task i with a learning effect if assigned to position r
St_j	Time of workstation j
α	Learning effect
CT	Cycle time
P	Set of couple of task (i,k) in which task i is immediate predecessor of task k

AP_i	Set of all predecessors of task i , including non- immediate predecessor
sut_{ik}	Setup time when task k is operated after task i inside same workstation
NT_j	Number of task assigned to workstation j
EC	Equipment cost per piece in $/ unit time
$X_{ijr} \in [0,1]$	1 is task i is assigned to rth sequence at workstation j
$Y_{ikj} \in [0,1]$	1 if task i is operated immediately before task k at workstation j in the same or in the next cycle

3.3 Objective functions and constraints

Relations (2)–(4) represent the objective functions. While Relation (2) minimizes the cycle time (equivalently maximizing the production rate of the line), Relation (3) minimizes the total equipment cost, and Relation (4) minimizes the smoothness index that helps to distribute the tasks evenly as possible to the workstations. (Hamta et al., 2013). These 3 relations helps to form a mixed integer nonlinear programming model.

$$MinZ_1 = CT \tag{2}$$

$$MinZ_2 = \sum_{j=1}^{m} NT_j EC \left[\sum_{i=1}^{n} \sum_{r=1}^{Mn} t_{ir}^{-1} X_{ijr} \right] \tag{3}$$

$$MinZ_3 = \sqrt{\sum_{j=1}^{m} (CT - St_j)^2} \tag{4}$$

These multi-objective function are subjected to constraints which are listed as follows

Each task must be assign to only one sequence position in only one workstation

$$\sum_{j=1}^{m} \sum_{r=1}^{Mn} X_{ijr} = 1 \qquad i=1,2,3...,n \tag{5}$$

At least one task must be assigned to the existing workstation.

$$\sum_{i=1}^{n} \sum_{r=1}^{Mn} X_{ijr} \geq 1 \qquad j=1,2,3...,m \tag{6}$$

At most one assigned task in each sequence position inside each workstation.

$$\sum_{i=1}^{n} X_{ijr} \leq 1 \qquad j=1,2,3\ldots,m. \quad r=1,2,3\ldots.,Mn \tag{7}$$

Task should be assigned in ascending order of positions in sequencing each workstation.

$$\sum_{i=1}^{n} X_{ij.r+1} - \sum_{i=1}^{n} X_{ijr} \leq 0 \tag{8}$$

The precedence relations between the tasks are not violated.

$$\sum_{j=1}^{m}\sum_{r=1}^{Mn}\{Mn(j-1)+r\}X_{ijr} - \sum_{j=1}^{m}\sum_{r=1}^{Mn}\{Mn(j-1)+r\}X_{kjr} \leq 0 \qquad \forall(i,k)\in P \tag{9}$$

Operation time of each task must be between given lower and upper bound.

$$Lt_i \leq t_i \qquad t_i \leq Ut_i \qquad i=1,2,3\ldots,n \tag{10}$$

Sum of operation times with learning effect and the corresponding setup time in each workstation doesn't exceed the cycle time.

$$\sum_{i=1}^{n}\sum_{r=1}^{Mn} t_{ir} X_{ijr} + \sum_{i=1}^{n}\sum_{k=1}^{n} sut_{ik} Y_{ikj} \leq CT \qquad i=1,2,3\ldots,n \tag{11}$$

Operation time of task i with a learning effect if assigned to position r, where α is constant learning index.

$$t_{ir} = \sum_{j=1}^{m} t_i r^{\alpha} X_{ijr} \qquad i=1,2,3\ldots,n$$

$$Mn \; ; \alpha \leq 0 \qquad r=1,2,3\ldots \tag{12}$$

Number of task assigned to workstation j.

$$NT_j = \sum_{i=1}^{n} \sum_{r=1}^{Mn} X_{ijr} \qquad j=1,2,3\ldots,m \qquad (13)$$

Time of workstation j.

$$St_j = \sum_{i=1}^{n} \sum_{r=1}^{Mn} t_{ir} X_{ijr} + \sum_{i=1}^{n} \sum_{\substack{k=1 \\ i \neq k}}^{n} sut_{ik} Y_{ikj} \qquad j=1,2,3\ldots,m \qquad (14)$$

State of lower and upper bounds of operation time.

$$\begin{aligned} Ut_i &> Lt_i \\ Lt_i &\geq 0 \end{aligned} \qquad i=1,2,3\ldots,n \qquad (15)$$

Binary nature of variables X_{ijr} and Y_{ikj} .

$$\begin{aligned} X_{ijr} &\in \{0,1\} \\ Y_{ikj} &\in \{0,1\} \end{aligned} \qquad i= 1,2,3\ldots,n; \; j=1,2,3\ldots,m; r=1,2,3\ldots,Mn \qquad (16)$$

3.4 The evaluation mechanism

In order to measure the quality of each ALBP solution generated by the proposed algorithm, we need an evaluation function. For this purpose, the combination of min–max and weighting methods is used in this dissertation. This procedure was first presented by Coello and Christiansen (1995) for a multi-objective optimization design problem. In the min–max technique, a solution is obtained while all the objectives are treated on terms of equal importance. When the min–max technique is combined with the weighting method, various Pareto (non-dominated) solutions can be generated. The evaluation function value is defined as follows, which aims to be minimized.

$$ZZ = Z_{p,w} = \left[\sum_{i=1}^{3} w_i \left(\frac{Z_i - Z_i^*}{Z_i^*} \right)^p \right]^{1/p} \qquad (17)$$

Where Zi^* and w_i denote the individual minima value and the weight of the ith objective function

$(0 \le w_i \le 1)$, respectively. The exponent p indicates the various ways of computing the scalarized distance between each objective and its optimal value. The most commonly used values for p are: 1 for the simplest formulation, 2 for the Euclidian distance, and ∞ for Tchebycheff norm (Behnamianetal., 2009). Since the scales of the objectives are different in our addressed problem, we will set p=1 to normalize the objective values.

3.4.1 Determining Z_i^*

In order to determine the best value of each objective function, the algorithm is run and from the iteration tables the minimum value is selected as Z_i^* (i= 1,2,3) for each objective function. The obtained values are replaced in Relation (17).

3.4.2 Determining the weight w_i

The weights w_i in Relation (17) indicate the relative importance of the corresponding objectives. The difficulty of the weighting method is the determination of appropriate values for the w_is. We found two general methods to calculate the weights in the literature of MO optimization problems (when there are more than two objectives): the fixed-weighted method and the random-weighted method. Given P objective functions, the first method uses fixed weight values with the following constraint

$$\sum_{i=1}^{P} w_i = 1 \qquad W_i > 0, \quad i = 1,2,\ldots, P \tag{18}$$

In the second method, the weights w_i are calculated by Relation (19) in which R_i are non-negative random numbers.

$$w_i = \frac{R_i}{\sum_{i=1}^{P} Ri} \qquad i = 1,2,\ldots, P \tag{19}$$

In comparison with the fixed-weighted method, the random- weighted method has the advantage that it gives the algorithm a tendency to show a variable search direction, enabling it to sample the search space uniformly to obtain a variety of non-dominated solutions (Gen and Cheng, 2000).

CHAPTER 4

ALBP MODEL OPTIMIZATION USING TLBO

4.1 TLBO Introduction

TLBO is the simulation of a classical school learning process proposed by Rao et al. (2012) that consist of two stages. During the first stage, called Teacher Phase, a teacher imparts knowledge directly to his/her students. The better the teacher, the more knowledge the students obtain. However, the possibility of a teacher's teaching being successful during the Teacher Phase, in practice, is distributed under Gaussian law. There are only very rare students who can understand all the materials presented by the teacher (i.e., the right end of the Gaussian distribution). Most students will partially accept new learning materials (i.e., the mid part of the Gaussian distribution) and, in some cases, the teacher will have almost no direct effect on students' knowledge (i.e., the left end of the Gaussian distribution). However, the possibility for most students to obtain new knowledge is not completely lost. During the second stage, called Learner Phase, a student may learn with the help of fellow students. Overall, how much knowledge is transferred to a student does not only depend on his/her teacher but also on interactions amongst students through peer learning.

4.1.1 TLBO methodology

TLBO methodology consists of two phases, namely, Teacher phase and Learner phase. These two phases are described below with the help of its pictorial representation.

Teacher Phase

It is first part of the algorithm where learners learn through the teacher. During this phase a teacher tries to increase the mean result of the class room from any value M_1 to his or her level. But practically it is not possible and a teacher can move the mean of the class room M_1 to any other value M_2 which is better than M_1 depending on his or her capability. Figure 2 shows the pictorial representation of how teacher tries to bring the level of class up to his or her level. Considered M_j be the mean and T_i be the teacher at any iteration i. Now T_i will try to improve existing mean M_j towards it so the new mean will be T_i designated as M_{new} and the difference between the existing mean and new mean is given by equation (20).

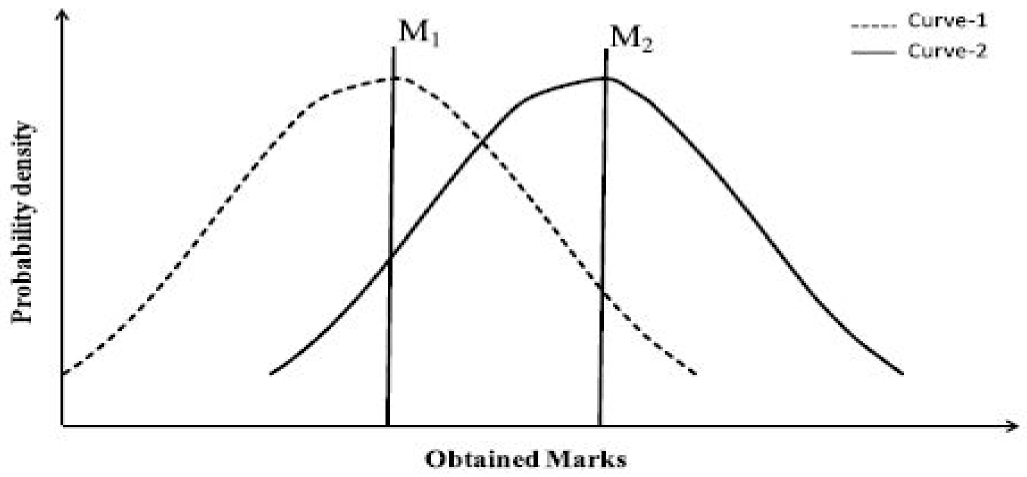

Figure 2: Pictorial demonstration of Teacher phase (Source: Rao et al., 2012)

$$Difference\_mean_i = r_i(M_{new} - T_F M_j) \tag{20}$$

Where T_F is a teaching factor that decides the value of mean to be changed and r_i is a random number in the range [0, 1]. The value of T_F can be either 1 or 2, which is again a heuristic step and decided randomly with equal probability as given in Equation (21).

$$T_F = round\ [1+ rand\ (0,1)\{2-1\}] \tag{21}$$

The teaching factor is generated randomly during the algorithm in the range of 1-2, in which 1 corresponds to no increase in the knowledge level and 2 corresponds to complete transfer of knowledge. The in between values indicates amount of transfer level of knowledge. The transfer level of knowledge can be any depending on the learners' capabilities. In the present work, attempt was carried out by considering the values in between 1-2, but any improvement in the results was not observed. Hence to simplify the algorithm the teaching factor is suggested to take either 1 or 2 depending on the rounding up criteria. However, one can take any value of T_F in between 1-2.Based on this *Difference_mean*, the existing solution is updated according to Equation (22).

$$X_{new,i} = X_{old,i} + Difference\_mean_i \tag{22}$$

Learner Phase

It is second part of the algorithm where learners increase their knowledge by interaction among themselves. A learner interacts randomly with other learners for enhancing his or her knowledge. A learner learns new things if the other learner has more knowledge than him or her.

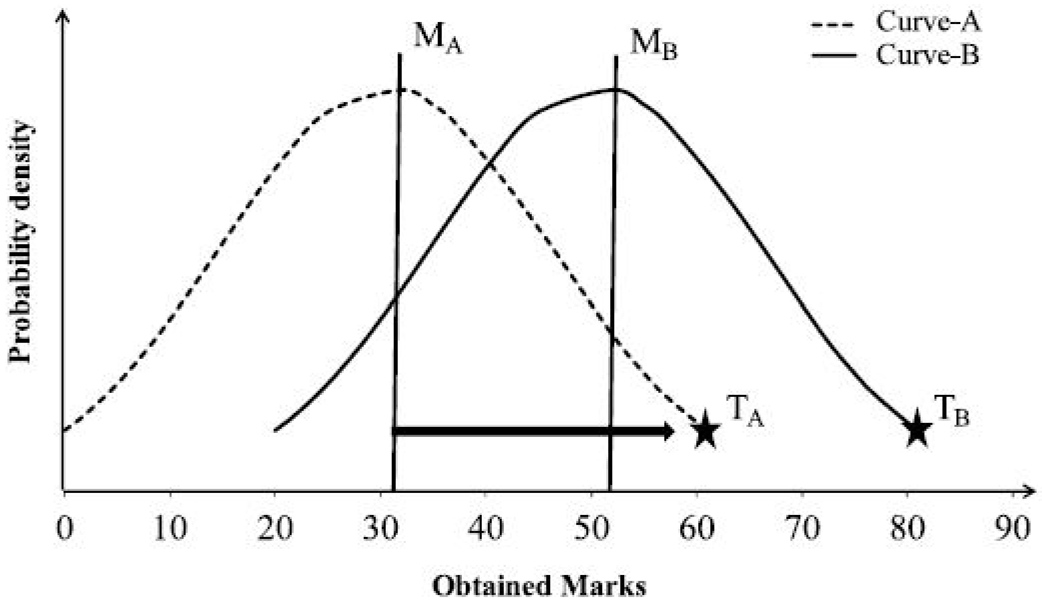

Figure 3: Pictorial demonstration of Learner phase (Source: Rao et al., 2012)

In Figure 3, it shows that the mean of the class shifted from M_A to M_B by interaction between the students itself. T_A is supposed to be the best solution for curve A, which is now shifted to T_B and the new curve for the class is curve B with mean M_B. Mathematically the learning phenomenon of this phase is expressed in Equation (23).

At any iteration *i*, considering two different learners *Xi* and *Xj* where $i \neq j$

$$X_{new,i} = X_{old,i} + r_i(X_i - X_j) \quad if\, f(X_i) < f(X_j) \tag{23}$$

$$X_{new,i} = X_{old,i} + r_i(X_j - X_i) \quad if\, f(X_j) < f(X_i)$$

Accept X_{new} if it gives better function value.

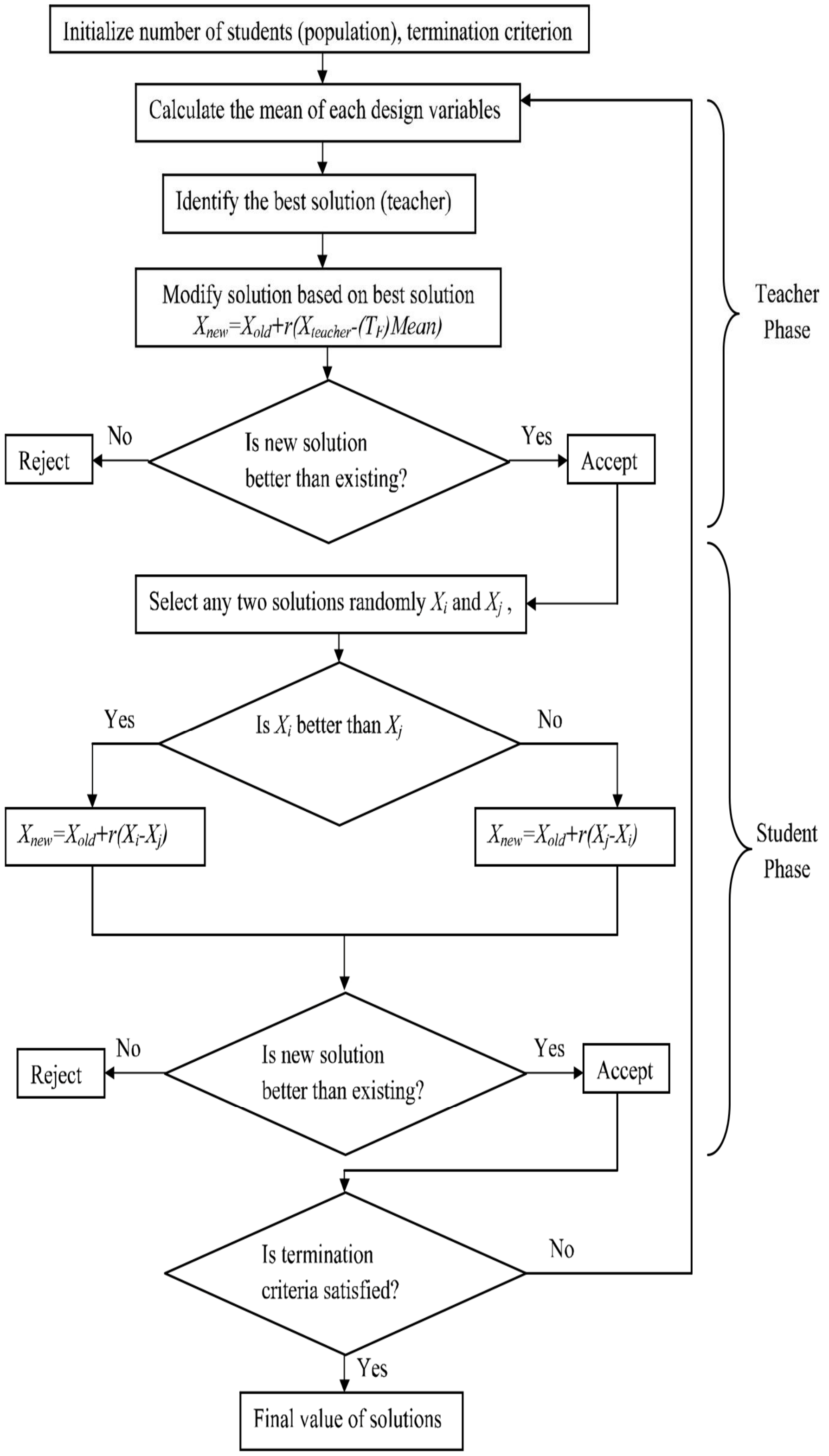
Initialize number of students (population), termination criterion
Calculate the mean of each design variables
Identify the best solution (teacher)
Modify solution based on best solution
$X_{new}=X_{old}+r(X_{teacher}-(T_F)Mean)$
Is new solution better than existing?
No
Reject
Yes
Accept
Teacher Phase
Select any two solutions randomly X_i and X_j ,
Is X_i better than X_j
Yes
No
$X_{new}=X_{old}+r(X_i-X_j)$
$X_{new}=X_{old}+r(X_j-X_i)$
Is new solution better than existing?
No
Reject
Yes
Accept
Student Phase
Is termination criteria satisfied?
No
Yes
Final value of solutions

Figure.4 Flow chart of TLBO (source: Rao et al., 2012)

4.2 ALBP model optimization using TLBO

For optimizing the ALBP model using TLBO, there will be need of some input data and steps required to attain the result. Followings are different sub sections which gives the input and step by step procedure for TLBO.

4.2.1 Data setting

The required data for our problem includes: the number of tasks, the number of workstations, precedence constraints between tasks, the lower and upper bounds of operation time of each task, setup times matrix and learning effect. The above required data, standard operation times and the precedence constraints for all problems are taken from a research paper (Hamta et al., 2013). Since this dissertation considers flexible operation times, a procedure is required to determine the lower and upper bounds of operation time of each task i (i,e Lt_i and Ut_i where i = 1,2,3…n). For this purpose, the times taken from the homepage are considered as Lt_is ($Ut_i = Lt_i + 2$). (Hamta et al., 2011). In this dissertation work the problems for 5 given number of workstations depending on the number of tasks (n) of the problem. Learning rates are chosen from the set of (70%, 80%). The setup times are also generated randomly using uniform distribution from 1 to the average of lower and upper bounds of operation times, i.e.

$$sut_{ik} = \begin{cases} \sim uniform \quad (1, \sum_{i=1}^{n} ((Lt_i + Ut_i)/2)/n) & i \neq k \\ 0 & i = k \end{cases} \tag{24}$$

Figure 5, shows the precedence relationship graph between different task and their operational time. It is required to understand the relations between the occurrences of the tasks one after another along with the time taken to complete the task. Its helps fulfill the constraints related to precedence of the tasks. There are 9 tasks considered for this situation, the circle shows the task number and square indicates the operational time for the respective task.

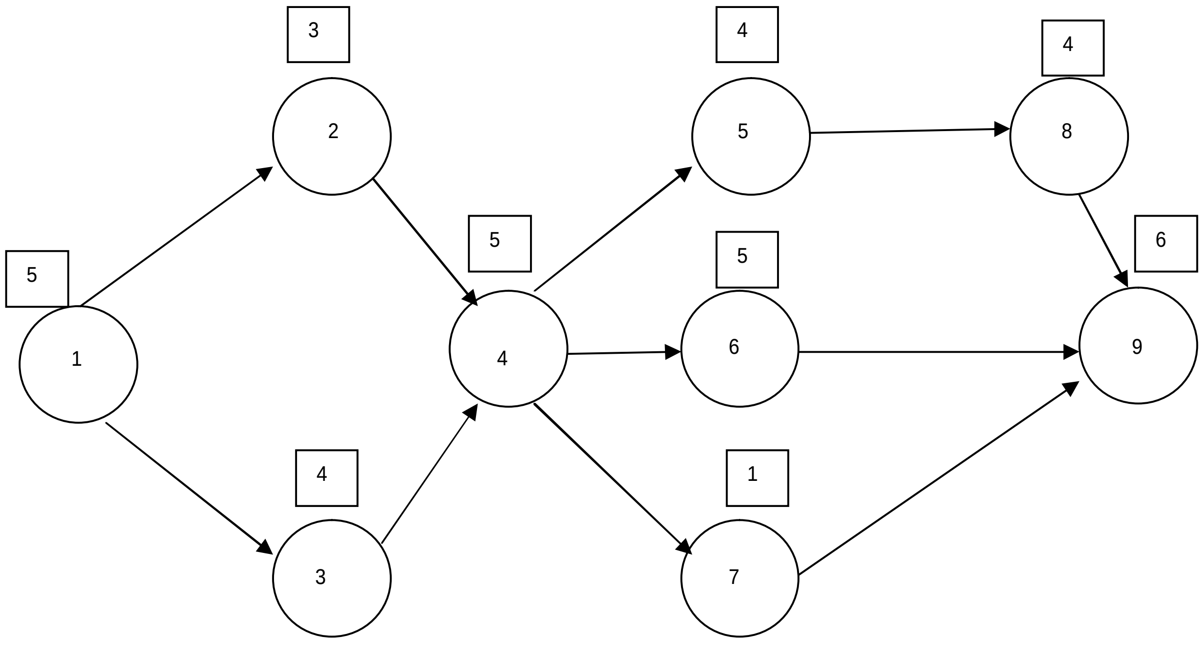

Figure 5: Precedence relationship of the input task

Cycle time

$$CT = \frac{\sum_{i=1}^{n} \hat{t}_i + \sum_{\substack{i,k \in R \\ i \neq k}} Sut_{ik}}{m} \tag{25}$$

Set the initial theoretical cycle time as CT (Hamta et al., 2012).where t_i is the average of upper time and lower time bound for each task i. Sut_{ik} is the setup time engaged if task k is operated on the same workstation after task i.

Station time

Station time is calculated by simply assign the task to workstation taking care of precedence relationships and CT. For example, taking 4 workstations and number of tasks 9 (Hamta et al., 2012). Number of task assigned to workstations and stations time are as shown in table.

Table 2: Assignment of task to workstation

Workstation	Task Assigned	Station Time
Station 1	1,2	5+3=8
Station 2	3,4,7	4+5+1=10
Station 3	5,6	5+4=9
Station 4	8,9	4+6=10

4.2.2 Initialization and assumptions

In this section, initialization for the problem of ALBP has done using some assumptions. The objective functions are already defined in the mathematical modeling section of the dissertation. Equations (2), (3) and (4) are used as objective functions and they are subjected to the constraints which are defined in Equations (5) to (16). The evaluation mechanism used for evaluating the difference of the objective function value to the minimum value possible, is from Equation (17).

This dissertation work has assumed that there is population size of 10 learners and termination criteria for the algorithm is set to be 10 iterations. Number of decision variables, setup time, number of workstations, upper and lower bound of time all are taken from research paper by Hamta et al., (2013

4.3 Steps of TLBO

To solve the ALBP using TLBO, the procedure is illustrated in 5 steps which are as follows

Step 1: Define the optimization problem and initialize the optimization parameters

Initialize of population size = 10

Number of iterations = 10

Number of decision variables = 9

Lower limit of decision variables = [5 3 4 5 4 5 1 4 6]

Upper limit of decision variables = [7 5 6 7 6 7 3 6 8] (Ut$_i$ = Lt$_i$ + 2). (Hamta et al., 2011).

In this step, the number of learners are taken as 10 and the subjects they are going to learn is 9 (decision variables). After 10 iterations the TLBO operation stop as 10 iterations is its termination criteria. Lower limit and upper limit of time of decision variables are taken from the research paper.

Step 2: Generation of initial population

Generate random population according to the population size and the number of decision variables. For TLBO, population size indicates the number of learners and the decision variables indicate the subjects (i.e. courses) offered. This population is expressed as

$$\text{Initial population} = \begin{bmatrix} x_{1,1} & x_{1,2} & . & . & x_{1,D} \\ x_{2,1} & x_{2,1} & . & . & x_{2,D} \\ . & . & . & . & . \\ . & . & . & . & . \\ x_{P_n,1} & x_{P_n,2} & . & . & x_{P_{n,D}} \end{bmatrix}$$

The initial population generated for each decision variable in tabulated below along with the values of the objective function *ZZ* (see table 2). Each decision variable is treated as a subject in this algorithm.

Step 3: Teacher Phase

The mean of the population generated for each decision variable is calculated and is presented in Equation no 26.

Mean (M_D) = [6.090609, 3.974634, 5.243629, 5.760552, 5.168927, 6.136196, 1.643742, 5.099559, 7.109908] (26)

The best solution amongst the learners is treated as a teacher in the teachers phase. In this example the best solution is given by Learner 10 and is presented in the Equation no 27.

Teacher = $X_{ZZ = \min}$ = [6.18379, 3.485254, 5.871885, 6.594359, 5.105295, 5.256981, 1.977176, 5.145025, 7.226284] (27)

The teacher now tries to shift the mean of the class according to the equation 20. The value of teaching factor is randomly assumed as 1 or 2. This obtained difference is added to the current population to update its values using equation (20)

The next step is to accept all the new modified solutions which give a better function value. The old solutions are replaced by new ones in this case and the other solutions are carried forward as it is. The new population at the end of Teachers phase is now tabulated in Table 4.

Step 4: Learner Phase

As already explained previously in this Chapter, the learners increase their knowledge by mutual interaction. Two learners from the population at the end of Teachers Phase are randomly selected and modified population is generated by using the equation (23).

The next step is similar to that of the teachers phase. Accept all the new modified solutions which give a better function value. The old solutions are replaced by new ones in this case and the other solutions are carried forward as it is. The new population at the end of Learners phase is now tabulated in Table 5.

Table 3: Generation of initial population for 1st iteration

Learner	x1	x2	x3	x4	x5	x6	x7	x8	x9	Z1	Z2	Z3	ZZ
1	6.180105	4.568153	5.492572	5.144468	4.700434	5.663198	1.33455	5.173216	7.381035	10.92755	7988.593	5.465538	0.041387
2	6.94444	4.675041	4.943015	5.802534	4.645796	6.595082	1.644449	5.994404	6.577331	11.36442	7087.362	5.472997	0.031865
3	5.1114	3.779343	4.13196	5.481204	4.761444	6.161892	1.142524	5.264013	6.632223	10.2932	8971.207	5.697746	0.057668
4	6.153199	3.504294	4.858865	5.058096	5.110835	5.908082	1.305856	4.365925	7.510784	10.55519	8362.012	5.761697	0.052725
5	5.854141	4.924837	5.600153	5.994554	4.974249	5.682433	2.175214	4.675758	7.598501	11.29597	6308.514	5.798092	0.029389
6	5.698886	3.863802	4.704628	5.016735	5.489256	6.729557	1.498266	5.259571	7.415311	10.9352	7661.072	6.359202	0.063596
7	6.519065	3.919517	5.235156	6.956582	5.652457	6.685243	2.383597	5.703722	7.034393	11.81795	5978.136	5.947995	0.033521
8	6.512811	4.004367	5.679115	5.647737	5.303331	6.294089	1.30667	4.932358	6.741326	11.08436	8009.033	5.283279	0.037545
9	5.748252	3.021737	5.918941	5.909253	5.946176	6.385399	1.669119	4.481597	6.981897	11.01247	7328.35	5.94252	0.046639
10	6.18379	3.485254	5.871885	6.594359	5.105295	5.256981	1.977176	5.145025	7.226284	11.16921	6693.98	5.281561	0.018436
Mean	6.090609	3.974634	5.243629	5.760552	5.168927	6.136196	1.643742	5.099559	7.109908	11.04555	7438.826	5.701063	0.041277

Table 4: Teachers phase for 1st iteration

Learner	x1	x2	x3	x4	x5	x6	x7	x8	x9	Z1	Z2	Z3	ZZ
1	6.250621	4.304772	5.616508	5.53687	4.68556	5.209328	1.580223	5.180464	7.437917	10.96045	7378.815	5.36563	0.101263
2	6.970882	4.257556	5.134541	6.27971	4.63164	6.366007	1.823291	5.997475	6.580528	11.40833	6769.009	5.235547	0.073553
3	5.136953	3.38758	4.518812	6.133183	4.702886	6.103282	1.367231	5.279617	6.657587	10.45743	8158.781	5.542388	0.139522
4	6.173397	3.309649	4.907071	5.109192	5.090411	5.115784	1.397426	4.4109	7.589092	10.42058	8197.272	5.731079	0.15502
5	5.854354	4.459445	5.675805	6.316997	4.957892	5.397993	2.256078	4.681177	7.663856	11.25272	6262.528	5.663393	0.076952
6	5.708197	3.432129	4.828164	5.825331	5.469229	6.541354	1.638947	5.263337	7.452811	11.0319	7327.443	6.074115	0.155755
7	6.558565	3.648616	5.564373	7.340609	5.60661	6.138196	2.388053	5.708755	7.103622	11.81148	5996.86	5.622207	0.076551
8	6.572564	3.740788	6.211708	5.820561	5.28509	5.797037	1.512705	4.968626	6.831976	11.14821	7469.948	5.201064	0.098452
9	5.758311	2.933537	5.949784	6.002386	5.899015	5.579578	1.970738	4.521542	7.078807	10.93874	6915.895	5.97235	0.124584
10	6.198973	3.453986	5.995685	6.938762	5.060403	4.986437	2.050856	5.14701	7.266764	11.21978	6592.469	5.152208	0.052822
Mean	6.118282	3.692806	5.440245	6.13036	5.138873	5.7235	1.798555	5.11589	7.166296	11.06496	7106.902	5.555998	0.105447

Table 5: Learner phase for 1st iteration

Learner	X1	X2	X3	X4	X5	X6	X7	X8	X9	Z1	Z2	Z3	ZZ
1	6.106409	4.35127	6.008767	5.211677	4.732385	5.196537	1.522009	5.149433	7.648011	10.9853	7509.411	5.500473	0.10671
2	5.375408	4.038369	4.653111	6.231158	4.697361	6.252916	1.812297	5.29118	6.654539	10.80127	7015.687	5.68941	0.092652
3	5.850996	3.333637	4.69864	5.330606	4.798026	5.543399	1.375586	4.76042	7.211112	10.38048	8228.485	5.596599	0.129744
4	6.429586	2.691269	4.370015	4.695639	5.129741	4.999669	0.834459	4.378686	7.53088	10.01199	11097.1	5.633194	0.256063
5	5.767685	4.30613	5.575735	6.286786	5.389134	5.791373	2.018159	4.69963	7.529589	11.27284	6515.866	5.75123	0.088123
6	5.485114	3.25879	4.272059	4.985443	5.427174	6.557984	1.268608	5.103052	7.470264	10.5657	8544.061	6.355266	0.208764
7	6.56283	3.704996	5.794617	6.895651	5.387645	5.847166	2.063858	5.543173	7.095089	11.579	6396.932	5.322072	0.058531
8	6.314852	3.69596	6.152872	5.933032	5.331961	5.653565	1.731074	4.627991	6.898882	11.06804	7076.721	5.396419	0.08085
9	5.409677	2.679343	5.906268	5.626268	6.200592	5.971982	1.908265	4.327141	6.994541	10.80482	7144.442	6.35242	0.15007
10	6.243238	3.724122	5.681911	6.158077	4.759009	5.134925	1.762613	5.151565	7.327916	10.98867	7061.979	5.172375	0.060455
Mean	5.954579	3.578389	5.311399	5.735434	5.185303	5.694952	1.629693	4.903227	7.236082	10.84581	7659.069	5.676946	0.123196

Step 5: Termination criterion

The algorithm halts when the termination criteria is satisfied else the algorithm restarts from step 3. The criterion used in this example is the maximum number of generations. Progress of the optimization algorithm for one generation depicting the modifications in Teachers phase and the Learner phase is presented in Table 3,4 and 5. It is clearly observed from Tables that the value for ZZ of the objective function decreases as the algorithm progresses from Teachers phase to the Learner phase in the same generation of the optimizing algorithm, and thus guarantee the convergence in the algorithm.

Above mention 5 steps give results for each iteration. Only 1st iteration is shown, the next 9 iterations are done with the help of code written for TLBO implementation. Actual code for TLBO is made on the basis of pseudo code available which is shown in Appendix B.

CHAPTER 5

RESULT AND DISCUSSION

It is observed from the literature that use of a particular optimization method or modification in a particular optimization method suits well to only a number of problems. However, the same method or modification may not work well for the other problems. In this dissertation, assembly line of a manufacturing system is considered for the application of TLBO. The problem considered for this purpose is taken from manufacturing domain of engineering and results obtained are compared with the previous work done by other algorithm i,e HPSO (Hybrid Particle Swarm Optimization). The proposal of using any new optimization algorithm requires a check of that new algorithm for a wide variety of problems before drawing any general conclusion for the modification incorporated. To check the performance of the proposed algorithm, 12 constraints problem are considered in this dissertation. After performing TLBO operation on the problem, graphs are drawn to show how the new algorithm gives better optimized value for the problem and small deviation from the minimum value possible. Algorithm is coded in MATLAB R2012a and run on personal computer with 2.40GHz Intel® Core™ i3-3110M CPU with 4 GB RAM memory under a Microsoft Windows 7 ultimate environment.

5.1 Experimental result

To examine the efficiency and effectiveness of the proposed algorithm over a large set of benchmark ALBPs taken from the open literature, the proposed TLBO algorithm, for ALBP, performance is measured on two performance measuring criteria namely by finding the Relative Percentage Deviation (RPD) and comparing to HPSO of Hamta et al.(2013).

5.1.1 Performance measure of model solution

After computing the evaluation function of Relation (17) for each test problem using the algorithms, relative percentage deviation (RPD) in percentage is calculated by the following relation (Hamta et al., 2011; Roshanaei et al., 2009):

$$RPD\,(\%) = \frac{A\lg_{sol} - Min_{sol}}{Min_{sol}} \times 100 \tag{28}$$

Where Alg_{sol} is the objective value of algorithm for a given problem and Min_{sol} is the best solution obtained for each instance by the algorithm. An average RPD equal to 4% generated by a specific algorithm means this algorithm is 4% over the best obtained solution on average. As it is clear, lower RPD values are preferred.

In this dissertation, like HPSO, TLBO algorithm also uses the weighting method and the weights are calculated by relation (19). In TLBO, the weights are set as $w_1 = w_2 = 0.3$ and $w_3=0.4$, in which the first and second objective functions have the equal weight, since they are clearly inconsistent and the third objective function has a little more weight to help the tasks to be distributed evenly as possible to the workstations.

The problem is solved by using population size of 10 learners and after performing 10 iterations, best objective value between the obtained results is set to Min_{sol} in Relation (28). Afterwards, RPD measure is calculated for each solution and the average RPD is reported for each algorithm and instance.

Tables 6 to 11 show the results of experiments obtained by the proposed TLBO algorithm and HPSO for two learning rates (70% and 80%), each one grouped by n and m. These tables include the average RPD in percentage for both the algorithm on different instances. As can be seen, TLBO provides better results than HPSO in terms of RPD measure. Both TLBO and HPSO gives near about the same result but TLBO provide slightly better result than HPSO. These examinations are better demonstrated in Figures. 5 and 6 in which the fluctuation of the average RPD is displayed for two learning rates.

Table 6: Comparison of RPD% for TLBO and HPSO (n=9 and learning rate =70%). (HPSO source: Hamta et al. (2013))

			RPD (%)	
Problem Name	Number of tasks (n)	Number of Stations (m)	TLBO	HPSO
DATA SET 1	9	4	0.946	1.576
	9	5	2.049	3.011
	9	6	5.222	4.885
	9	7	1.425	1.956
	9	8	1.736	2.043
			2.275	2.6942

Table 7: Comparison of RPD% for TLBO and HPSO (n=30 and learning rate =70%). (HPSO source: Hamta et al. (2013))

			RPD (%)	
Problem Name	Number of tasks (n)	Number of Stations (m)	TLBO	HPSO
DATA SET 2	30	7	1.005	1.076
	30	8	1.097	1.141
	30	9	1.29	1.297
	30	10	1.001	1.001
	30	11	1.016	1.022
			1.0818	1.1074

Table 8: Comparison of RPD% for TLBO and HPSO (n=45 and learning rate =70%). (HPSO source: Hamta et al. (2013))

			RPD (%)	
Problem Name	Number of tasks (n)	Number of Stations (m)	TLBO	HPSO
DATA SET 3	45	9	0.887	0.902
	45	10	0.901	0.967
	45	11	0.817	0.881
	45	12	0.832	0.894
	45	13	0.921	0.979
			0.8716	0.9246

Table 9: Comparison of RPD% for TLBO and HPSO (n=9 and learning rate =80%). (HPSO source: Hamta et al. (2013))

			RPD (%)	
Problem Name	Number of tasks (n)	Number of Stations (m)	TLBO	HPSO
DATA SET 1	9	4	0.785	1.203
	9	5	0.376	0.954
	9	6	2.155	2.517
	9	7	3.645	3.856
	9	8	1.581	2.011
			1.709	2.1082

Table 10: Comparison of RPD% for TLBO and HPSO (n=30 and learning rate =80%). (HPSO source: Hamta et al. (2013))

			RPD (%)	
Problem Name	Number of tasks (n)	Number of Stations (m)	TLBO	HPSO
DATA SET 2	30	7	0.619	0.823
	30	8	0.553	0.78
	30	9	0.892	0.901
	30	10	0.85	0.894
	30	11	0.77	0.779
			0.7368	0.8354

Table11: Comparison of RPD% for TLBO and HPSO (n=45 and learning rate =80%). (HPSO source: Hamta et al. (2013))

			RPD (%)	
Problem Name	Number of tasks (n)	Number of Stations (m)	TLBO	HPSO
DATA SET 3	45	9	0.784	0.797
	45	10	0.787	0.797
	45	11	0.84	0.843
	45	12	0.856	0.879
	45	13	0.891	0.89
			0.8316	0.8412

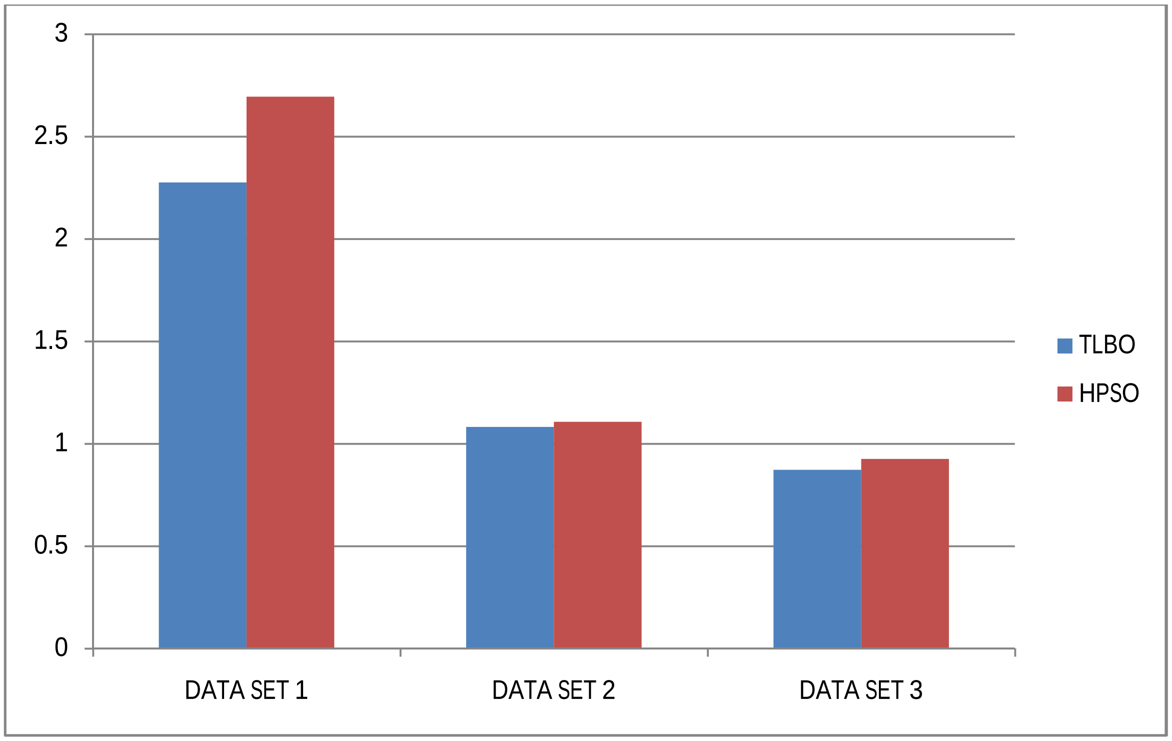

Figure.5: Comparative results for TLBO and HSPO (learning rate =70%)

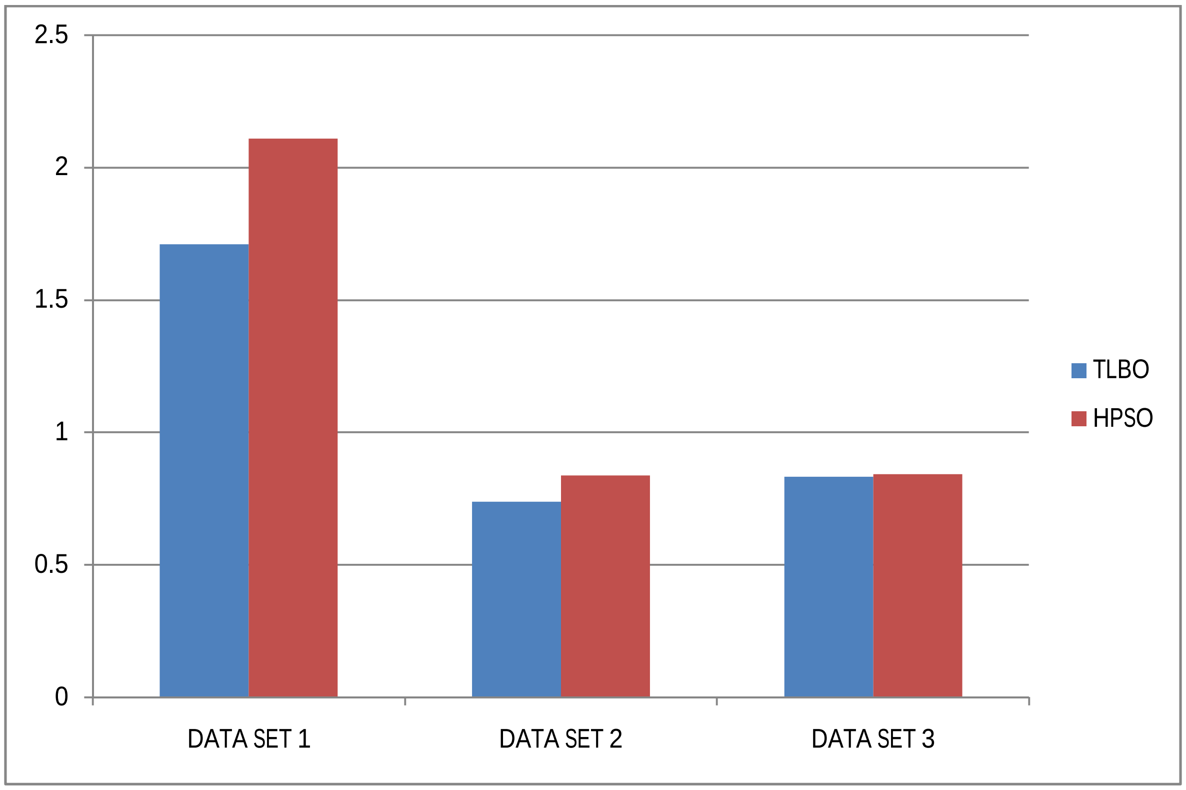

Figure.6: Comparative results for TLBO and HPSO (learning rate =80%)

CHAPTER 6

CONCLUSION

6.1 Summary

The competitive world today demands effective and efficient equipment and machinery which necessitates optimization to be carried out at every stage from conceptualization to manufacturing to reduce the cost and proper utilization of scarce resources. Although non-traditional optimization techniques have been used in the past to solve optimization problems in both design and manufacturing domain but these algorithms have their own limitations and drawbacks. Evolutionary algorithms can effectively address some of the limitations of traditional algorithms; hence widely applied in various fields of engineering with varying degree of success. The quality of solutions generated by these algorithms is highly dependent on the tuning of algorithmic parameters. All evolutionary algorithms such as GA, SA, ABC and others require algorithm specific parameters in addition to the common parameters of population size and number of generations. A change in these algorithmic parameters changes the overall effectiveness of the algorithm. To avoid this difficulty, a population based optimization algorithm and its improved versions are presented in this dissertation and have been applied to optimization problems of manufacturing domain.

This work dealt with a multi-objective single-model ALBP when task operation time is between the lower and upper bounds and sequence-dependent setup times exist between the tasks. We assumed that the task time is dependent on worker(s) (or machine(s)) learning for the same activity. Three objectives were considered: (1) minimizing the cycle time (equivalently maximizing the production rate), (2) minimizing the total equipment cost, and (3) minimizing the smoothness index. We also proposed well-known test problems taken from the literature. The results indicated that our algorithm provide slightly better results than the solution given by HPSO for all objective functions. TLBO provides fast and better result as compared to any other evolutionary techniques. As it has less parameter than those traditionally used techniques, it provides much more flexibility in its use. RPD (%) values obtained in the results shows that the deviation of the optimum values, which satisfy all the objectives of multi objective model, not varying too much from the minimum value possible for each objective.

6.2 Future Research

This work can be enriched with other assumptions such as U-shaped, two-sided lines, parallel stations and equipment selection. On the other hand, development of an exact solution technique (e.g., branch and bound method) to solve relatively large-scale problems is a challenging area for future studies.

Akpınar, S., & Bayhan, G. M. 2011. A hybrid genetic algorithm for mixed model assembly line balancing problem with parallel workstations and zoning constraints. *Engineering Applications of Artificial Intelligence*, *24*(3), 449–457.

Andres, C., Miralles, C., Pastor, R., 2008. Balancing and scheduling tasks in assembly lines with sequence-dependent setup times. *European Journal of Operational Research* 187, 1212–1223.

Baykaso, A. 2014. Modeling and solving mixed-model assembly line balancing problem with setups . Part I : A mixed integer linear programming model. *Journal of manufacturing science*, *33*, 177–187.

Becker, C., Scholl, A., 2006. A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research* 168 (3), 694–715.

Biskup, D., 1999. Single-machine scheduling with learning considerations. *European Journal of Operational Research* 115, 173–178.

Boysen, N., Fliedner, M., Scholl, A., 2007. A classification of assembly line balancing problems. *European Journal of Operational Research* 183, 674–693.

Bukchin, Y., Rabinowitch, I., 2006. A branch-and-bound based solution approach for the mixed-model assembly line-balancing problem for minimizing stations and task duplication costs. *European Journal of Operational Research* 174, 492–508.

Cakir, B., Altiparmak, F., Dengiz, B., 2011. Multi-objective optimization of a stochastic assembly line balancing: a hybrid simulated annealing algorithm. *Computers & Industrial Engineering* 60, 376–384.

Cohen, Y., Vitner, G., Sarin, S.C., 2006. Optimal allocation of work in assembly lines for lots with homogenous learning. *European Journal of Operational Research* 168 (3), 922–931.

Ege, Y., Azizoglu, M., Ozdemirel, N.E., 2009. Assembly line balancing with station paralleling. *Computers & Industrial Engineering* 57, 1218–1225.

Erel, E., Sarin, S.C., 1998. A survey of the assembly line balancing procedures. *Production Planning & Control* 9 (5), 414–434.

Gamberini, R., Grassi, A., Rimini, B., 2006. A new multi-objective heuristic algorithm for solving the stochastic assembly line re-balancing problem. *International Journal of Production Economics* 102, 226–243.

Ghosh, S., Gagnon, R.J., 1989. A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems. *International Journal of Production Research* 27 (4), 637–670.

Gutjahr, A.L., Nemhauser, G.L., 1964. An algorithm for the line balancing problem. *Management Science* 11 (2), 308–315.

Hamzadayi, A., & Yildiz, G. 2013. A simulated annealing algorithm based approach for balancing and sequencing of mixed-model U-lines q. *Computers & Industrial Engineering*, *66*(4), 1070–1084.

Hamzadayi, A., & Yildiz, G. 2012. A genetic algorithm based approach for simultaneously balancing and sequencing of mixed-model U-lines with parallel workstations and zoning constraints. *Computers & Industrial Engineering*, *62*(1), 206–215.

Henig, M.I., 1986. Extensions of the dynamic programming method in the deterministic and stochastic assembly-line balancing problems. *Computers & Operations Research* 13, 443–449.

Kara, Y.,Ozguven, C.,Sec,me, N.Y.,Chang,C.T.,2011.Multi-objective approaches to balance mixed-model assembly lines for model mixes having precedence conflicts and duplicable common tasks. *International Journal of Advanced Manufacturing Technology* ,52,725–737.

Kim, Y. K., Song, W. S., & Kim, J. H. 2009. A mathematical model and a genetic algorithm for two-sided assembly line balancing, *Computers and operational research, 36*, 853–865.

Kottas, J.F., Lau, H.S., 1981. A stochastic line balancing procedure. *International Journal of Production Research* 19, 177–193.

Lin et al. 2010. An efficient job shop scheduling algorithm based on particle swarm optimization. *Expert Systems with Applications* 37, 2629–2636.

Manavizadeh, etal. 2013. A Simulated Annealing algorithm for a mixed model assembly U-line balancing type-I problem considering human efficiency and Just-In-Time approach. *Computers & Industrial Engineering*, *64*(2), 669–685.

Mather, H., 1989. Competitive manufacturing. Prentice Hall, Englewood Cliffs, NJ.

Meyr, H., 2004. Supply chain planning in the German automotive industry. OR Spectrum 26, 447–470.

Miralles, C.,Garcıa-Sabater, J.P.,Andres, C.,Cardos, M.,2008. Branch and bound procedures for solving the assembly line worker assignment and balancing problem: application to sheltered work centers for disabled. *Discrete Applied Mathematics* 156,352–367.

Mosheiov, G., 2001. Scheduling problems with a learning effect. *European Journal of Operational Research* 132, 687–693.

Nearchou, A.C., 2007. Balancing large assembly lines by a new heuristic based on differential evolution method. *International Journal of Advanced Manufacturing Technology* 34, 1016–1029.

Nicosia, G., Pacciarelli, D., Pacifici, A., 2002. Optimally balancing assembly lines with different workstations. *Discrete Applied Mathematics* 118, 99–113.

Nourmohammadi, A., Zandieh, M., 2011. Assembly line balancing by a new multi- objective differential evolution algorithm based on TOPSIS. *International Journal of Production Research* 49, 2833–2855.

Özbakır, L., & Tapkan, P. 2011. Bee colony intelligence in zone constrained two-sided assembly line balancing problem. *Expert Systems With Applications*, *38*(9), 11947–11957.

Pine, B.J., 1993. Mass customization: The new frontier in business competition. Harvard Business School Press, Boston, Mass. Journal of Manufacturing and Operations Management 3, 200–223.

Polat, O., & Supciller, A. A. 2013. An iterative genetic algorithm for the assembly line worker assignment and balancing problem of type-II, *Computers & Operations Research .40*, 418–426.

Ponnambalam, S.G., Aravindan, P., Mogileeswar Naidu, G., 2000. A multi-objective genetic algorithm for solving assembly line balancing problem. International *Journal of Advanced Manufacturing Technology* 16, 341–352.

Ponnambalam, S.G., Aravindan, P., Mogileeswar Naidu, G., 2000. A multi-objective genetic algorithm for solving assembly line balancing problem. *International Journal of Advanced Manufacturing Technology* 16, 341–352.

R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan (1995) 39–43.

Ramezanian, R., & Ezzatpanah, A. 2015. Computers & Industrial Engineering Modeling and solving multi-objective mixed-model assembly line balancing and worker assignment problem. *Computers & industrial engineering*, *87*, 74–80.

Rao and Patel, 2012.An elitist teaching learning based algorithm for solving complex constrained optimization problem. *International Journal of Industrial Engineering Computations*, *3*, 535–560.

Roshani, A., Roshani, A., Roshani, A., Salehi, M., & Esfandyari, A. 2013. A simulated annealing algorithm for multi-manned assembly line balancing problem. *Journal of Manufacturing Systems, 32*(1), 238–247.

Rubinovitz, J., Levitin, G., 1995. Genetic algorithm for assembly line balancing. International *Journal of Production Economics* 41, 343–354.

Sabuncuoglu, I., Erel, E., & Alp, A. 2009. Ant colony optimization for the single model U-type assembly line balancing problem. *International Journal of Production Economics*, *120*(2), 287–300.

Scholl, A., Klein, R., 1999. ULINO: Optimally balancing Ushaped JIT assembly lines. International Journal of Production Research 37, 721–736.

Scholl, A., Becker, C., 2005. A note on an exact method for costoriented assembly line balancing. International Journal of Production Economics 97, 343–352.

Scholl, A., Becker, C., 2006. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. European Journal of Operations Research 168, 666–693.

Seyed-Alagheband, S.A., Fatemi Ghomi, S.M.T., Zandieh, M., 2011. A simulated annealing algorithm for balancing the assembly line type II problem with sequence-dependent setup times between tasks. *International Journal of Production Research* 49 (3), 805–825.

Simaria, A. S., & Vilarinho, P. M. 2004. A genetic algorithm based approach to the mixed-model assembly line balancing problem of type II. *Computers and industrial engineering*, *47*, 391–407.

Simaria, A.S., Vilarinho, P.M., 2009. 2-ANTBAL: an ant colony optimization algorithm for balancing two-sided assembly lines. *Computers & Industrial Engineering* 56, 489–506.

Shtub, A., Dar-El, E.M., 1989. A methodology for the selection of assembly systems. International Journal of Production Research 27, 175–186.

Shtub, A., Dar-El, E.M., 1990. An assembly chart oriented assembly line balancing approach. International Journal of Production Research 6, 1137–1151.

Taha, R. B., El-kharbotly, A. K., Sadek, Y. M., & Afia, N. H. 2011. A Genetic Algorithm for solving two-sided assembly line balancing problems. *Ain Shams Engineering Journal*, *2*(3-4), 227–240.

Tapkan, P., Ozbakir, L., & Baykasoglu, A. 2012. Modeling and solving constrained two-sided assembly line balancing problem via bee algorithms. *Applied Soft Computing Journal*, *12*(11), 3343–3355.

Toksarı, M.D.,Isleyen,S.K.,Guner, E.,Baykoc, O.F.,2008. Simple and U-type assembly line balancing problems with a learning effect. *Applied Mathematical Modelling*,32,2954–2961.

Toksarı, M.D.,Isleyen,S.K.,Guner, E.,Baykoc, O.F.,2010. Assembly line balancing problem with deterioration tasks and learning effect. *Expert Systems with Applications* 37,1223–1228.

Tiacci, L. 2015. Coupling a genetic algorithm approach and a discrete event simulator to design mixed-model un-paced assembly lines with parallel workstations and stochastic task times. *International Journal of Production Economics*, *159*, 319–333.

Yolmeh, A., & Kianfar, F. 2012. An efficient hybrid genetic algorithm to solve assembly line balancing problem with sequence-dependent setup times. *Computers & Industrial Engineering*, *62*(4), 936–945.

Yuan, B., Zhang, C., Shao, X., & Jiang, Z. 2015. An effective hybrid honey bee mating optimization algorithm for balancing mixed-model two-sided assembly lines. *Computers and Operation Research*, *53*, 32–41.

Zha, J., & Yu, J. 2014. Technical paper A hybrid ant colony algorithm for U-line balancing and rebalancing in just-in-time production environment. *Journal of Manufacturing Systems*, *33*(1), 93–102.

Zhao, X., Hsu, C., Chang, P., & Li, L. 2015. A genetic algorithm for the multi-objective optimization of mixed-model assembly line based on the mental workload, *Engineering Applications of Artificial Intelligence* , 1–7.

APPENDIX A

Research Papers and its related information

Journal	Author	Year	AI Tech.	Field
Expert system and application	Hamid. S et al.	2009	Fuzzy Knowledge - based expert system and Genetic algorithm	Material Handling
International Journal of Machine Tools And Manufacture	Wang. K et al.	2003	ANN and Genetic Algorithm	Electro-Discharge Machining
Computer and Industrial Engineering	Zeidi. J. R et al.	2013	Genetic Algorithm and Neural Network	Cellular Manufacturing system
Applied Mathematical Modelling	Zhang. H et al.	2012	Artificial bee colony	Copper strip production
Computer in industry	Szelke. E and Markus. G	1997	SUPREACT	Reactive Scheduling
Engineering Application of Artificial Intelligence	Papa. G and Seljak. B.K	2005	Genetic Algorithm	designing
Computer Aided Design	Marchetta. M. G and Forradellas. R. Q	2010	AI planning	Process planning
Journal of Material Processing Technology	Kumar. S And Singh.R	2008	AutoLISP and AutoCAD	Strip layout design
Knowledge Based System	Shakya. S et al.	2010	Genetic Algorithm	
Artificial Intelligence In Engineering	McMullen. P .R	2001	Ant colony optimization, Simulated Annealing, Tabu search, Genetic algorithm, ANN	Logistic
European Journal of operational research	Wu. T. H et al.	2010	Water Flow Algorithm	Manufacturing
Journal of Material Processing Technology	Kim. H. S and Im. Y. T	1999	Depth first search	Cold forging
European Journal of operational research	Bautista. J and Pereira. J	2007	Ant colony optimization	time and space constrained assembly line balancing problem
Computer in industry	Barschdorff.D et al	1997	ANN and Fuzzy	Virtual Manufacturing
Material Scinece and Engineering	Guessasma. S et al	2004	Artificial Neural Network	Atmospheric Plasma Spray

				Process
Computer and operations research	In lee	2001	Tabu Search	Multi machinig two stage scheduling
Expert system with applications	Jang Hee Lee	2002	Artificial Neural Network	Dynamic sampling
Control Engineering practices	Valle. C. D and Camacho. E. F	1996	Algorithm A*	assembly task
Applied Soft Computing	Altiparmak. F et al	2007	Artificial Neural Network	asynchronous assembly systems (AAS).
International journal of production economics	Gunter Schmidt	1998	Case based reasoning	Production scheduling problem
Expert system with applications	Kim. K. S et al.	2008	adaptive learning network (ALN),	Small Manufacturing Enterprises
Engineering Application of Artificial Intelligence	Sinha. Ashesh K et al.	2012	Immuno- particle swarm optimization with penetrated hyper- mutation (COIPSO-PHM)	inventory replenishment
Engineering Application of Artificial Intelligence	Tamayo. S et al	2009	Genetic Algorithm and Neural Network	Deliveries optimization
Expert systems with Application	Su. C. T and Wong. J. T	2007	Particle Swarm Optimization and ANN	Designing MIMO controller
Machine tools and manufacture	Scheffer. C et al	2003	Neural Networks	Process Monitoring
Computers in industry	Han. T and Yang. B. S	2006	GA and ANN	e-maintenance
Computer industrial Engineering	shih. W and Srihari. K	1995	Distributed Artificial Intelligence (DAI) and Fuzzy	Process planning
Computer industrial Engineering	Chu. E et al	1996	Distributed Artificial Intelligence (DAI) and Fuzzy	Process planning
European journal of operational research	Kullkarni . U. R and Kiang. M. Y	1995	ANN	Decision Support system and Group technology
Engineering Application of Artificial Intelligence	Lou. H. H and Huang .Y .L	2003	Fuzzy logic	Automotive coating and quality control

Engineering Application of Artificial Intelligence	Jozefczyk. J	2006	Simulated annealing	Flexible Maufacturing System
Journal of Material Processing Technology	k.Cheng et al	1998	ANN and Fuzzy logic system	Agile manufacturing
Knowledge Based System	Neuroth. M et al	2000	ANN and Fuzzy logic	industrial process
Simulation Modeling Practice and Theory	orsoni. A and Bandinelli. R	2007	ANN	Scheduling
Robotics and computer integrated manufacturing	Hsieh. K-H	2010	ANN	Process analysis
Neurocomputing	Ficko. M at al	2010	Particle Swarm Optimization and Shortage travel path	Flexible Manufacturing System
Expert system with applications	Guh. R-S et al	1999	Hybrid AI (ANN + IntelliSPC tool)	Process control
Engineering Application of Artificial Intelligence	Banharnsakun. A et al	2012	Artificial bee colony	Job Shop Scheduling
Annual review in control	Monostori. L et al	1998	Proactive and reactive scheduling	distributed control architecture
Journal of Material Processing Technology	Dobrzanski. L. A et al	2005	ANN and GA	Forecasting
Applied Soft Computing	Tapkan .P et al	2012	Artificial bee colony algorithm	Two- sided assembly line balancing
Neurocomputing	Madureira. A et al	2014	Swarm intelligence	Dynamic scheduling
Surface and coating technology	Vitanov. V. I et al	2001	Neurofuzzy	Friction Surfacing
Journal of Material Processing Technology	Lorenzo. R. D et al	2006	ANN	Forming process design
Expert system with applications	Lolas. S and Olatunbosun. O. A	2008	Neural Networks	Reliability prediction
Neurocomputing	Noroozi. A et al	2013	ANN	Batch processing planning
computers in industrial engineering	Leu. Y. Y et al	1996	Genetic algorithm	assembly line sequencing
International journal of production economics	Chan. F. T. S et al	2000	ANN and fuzzy	FMS design

Engineering Application of Artificial Intelligence	Pacella. M and Semeraro. Q	2005	ANN	Statistical process control
International journal of production research	Yang. T and Lu. J. C	2010	ANN	Dispatching problems
journal of the Chinese institute of industrial engineers	Dao. T. M et al	2007	Neural Networks	Group scheduling
International journal of production research	Vieira. G. E and Ribas. P. C	2004	Simulated annealing	Master production scheduling
International journal of production research	Ransing. R. S and Lewis. R. W	1997	Neural Networks	Manufacturing dignosis
The journal of the textile institute	Wong. W. K and Chan. C. K	2009	Genetic algorithm	Production planning and scheduling
International Journal of computer integrated manufacturing	T cakar and I Cil	2004	ANN	Manufacturing system
International Journal of Production research	R. Piplani and J. Talavage	1997	Simulation and goal regression	Manufacturing system

APPENDIX B

Teacher Learner Based Optimization (TLBO) pseudo code

```
Set $k=1$;
Objective function $f(X)$,    $X=(x_1,x_2,\ldots\ldots,x_d)^T$        $d$=no. of design variables
Generate initial students of the classroom randomly  $X^i$, $i=1,2,\ldots,n$      $n$=no. of students
Calculate objective function $f(X)$ for whole students of the classroom
    WHILE (the termination conditions are not met)
        $\{Teacher\ Phase\}$
        Calculate the mean of each design variable $X_{Mean}$
        Identify the best solution (teacher)
        FOR $i=1\rightarrow n$
            Calculate teaching factor  $T_F^{\,i} = round\left[1+rand(0,1)\{2-1\}\right]$
            Modify solution based on best solution(teacher)  $X^i_{new} = X^i + rand(0,1)\left[X_{teacher} - (T_F^{\,i}\cdot X_{mean})\right]$
            Calculate objective function for new mapped student  $f(X^i_{new})$
            IF  $X^i_{new}$ is better than $X^i$, i.e $f(X^i_{new}) < f(X^i)$
                $X^i = X^i_{new}$
            END IF  $\{End\ of\ Teacher\ Phase\}$
            $\{Student\ Phase\}$
            Randomly select another learner $X^j$, such that $j \neq i$
            IF  $X^i$ is better than $X^j$, i.e $f(X^i) < f(X^j)$
                $X^i_{new} = X^i + rand(0,1)\left(X^i - X^j\right)$
            Else
                $X^i_{new} = X^i + rand(0,1)\left(X^j - X^i\right)$
            END IF
            IF  $X^i_{new}$ is better than $X^i$, i.e $f(X^i_{new}) < f(X^i)$
                $X^i = X^i_{new}$
            END IF  $\{End\ of\ Student\ Phase\}$
        END FOR
        Set  $k=k+1$
    END WHILE
Postprocess results and visualization
```

(Source: Baghlani and Makiabadi, 2013)