Assembly Line Balancing and Sequencing

Chapter · August 2011

DOI: 10.5772/19953 · Source: InTech

2 authors:

Kamal Uddin
Wärtsilä
7 PUBLICATIONS **92** CITATIONS
SEE PROFILE

Jose Luis Martinez Lastra
Tampere University
254 PUBLICATIONS **2,836** CITATIONS
SEE PROFILE

2

Assembly Line Balancing and Sequencing

Mohammad Kamal Uddin and Jose Luis Martinez Lastra
Tampere University of Technology
Finland

1. Introduction

Assembly line balancing (ALB) and sequencing is an active area of optimization research in operations management. The concept of an assembly line (AL) came to the fact when the finished product is inclined to the perception of product modularity. Usually interchangeable parts of the final product are assembled in sequence using best possibly designed logistics in an AL. The initial stage of configuring and designing an AL was focused on cost efficient mass production of standardized products. This resulted in high specialization of labour and the corresponding learning effects. However, the recent trend gained the insight of the manufacturers of shifting the AL configuration to low volume assembly of customized products, mass customization. The strategic shift took effect due to the diversified customer needs along with the individualization of products. This eventually triggered the research on AL balancing and sequencing for customized products on the same line in an intermix scenario, which is characterized as mixed-model assembly line balancing (MMALB) and sequencing. The configuration planning of such ALs has acquired an important concern as high initial investment is allied with designing, installing and re-designing an AL.

The research carried out in this manuscript aims to contribute to the problem domain of MMALB and sequencing. Balancing refers to objective depended workload balance of the assembly jobs to different workstations. Sequencing refers to find an optimal routing/job dispatching queue considering the demand scenario, available time slots and resources. Primary factors associated to this problem domain includes different assembly plans (e.g. mixed/batch/single model production), variations in processing workstations (e.g. manual/robotic/hybrid stations), physical line layouts (e.g. straight/parallel/U-shaped lines) and varied work transporting methods (e.g. conveyor/pallet-based). These factors are mostly plant specific and must be considered as the design pre-requisites for line balancing and sequencing.

The contribution of this work is twofold. Firstly, a brief review of the problem domain of ALB and sequencing is presented. This includes systematic design approach of an AL and different performance and workstation related indexes which helps the line designer to identify plant specific design factors for line balancing, re-balancing and sequencing. Different heuristics and meta-heuristics based ALB solution strategies, classification of ALB problems, MMALB and sequencing are also addressed (section 2).

Secondly, a logic and mathematical formulation based methodology for solving ALB problem is proposed (section 3), addressed to low volume product customization in shop

floor (MMALB). The presented methodology results in optimizing the shift time for any combination of product customization, assembled in an intermix order. It also defines a repetitive job dispatching queue in accordance to the balancing results. The proposed approach is encoded via MATLAB and validated with reference data to prove the optimal conditions. A small scale practical shop floor problem is also analysed with the presented methodology (section 4) to show the optimality conditions. The conclusions are drawn in section 5.

2. Assembly line design

Systematic design of ALs is not an independent and easy task for the manufacturers. Designers need to deal with current physical factory layout in the initial phase. Cost and reliability of the system, complexity of the tasks, equipment selection, ALs operating criteria, different constraints, scheduling, station allocation, inventory control, buffer allocation are the most important area of concern. The development of an approach to design of ALs consisting of seven phases depicted in figure 1.

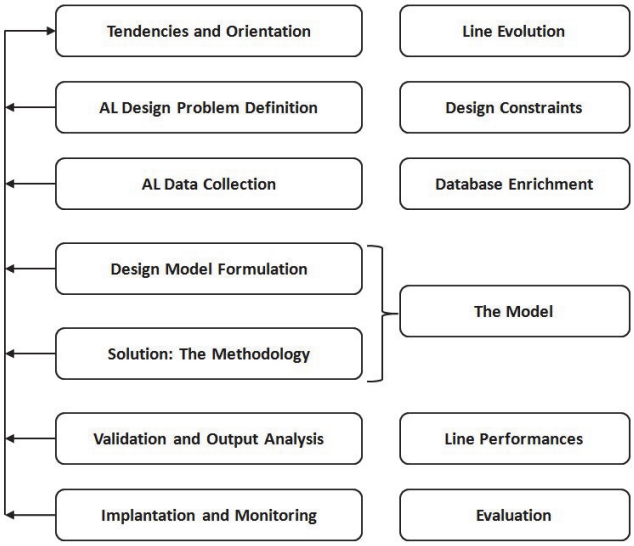

Fig. 1. Development of an approach to AL design (Rekiek & Delchambre, 2006)

Tendencies and orientation of ALs are linked to line evolution. Designers need to collect information in this step about the tendencies of the line to be implemented. Balancing and sequencing problem varies with the types of ALs. For instance, single model line produces a single product over the line. Facility layout, tool changes, workstation indexes remains fairly constant. Batch model lines produce small lots of different products on the line in batches. In mixed-model case, several variations of a generic product are produced at the same time in an intermixed scenario. Consideration of work transport system is also a concern. Apart from manual work transport on the line, three types of mechanized work transport systems are identified as continuous transfer, synchronous transfer (intermittent transfer) and

asynchronous transfer (Papadopoulos et al., 1993). Different line orientations need to be identified by the designer as it varies widely according to the production floor layout. Straight, Parallel, U-Shaped lines (Becker & Scholl, 2006) are generally applied.
Various design factors are important to study and integrate with the AL design and balancing. The decisive solution variations depend on the production approaches, objective functions and constraints. Some of the design constraints related to ALB are precedence constraints, zoning constraints and capacity constraints (Vilarinho & Simaria, 2006). Efficient description of line design problem is associated with database enrichment. To collect AL data, knowledge about several performance indexes and workstation indexes are important for a line designer (Table 1).

Performance Indexes	Workstation Indexes
1. Variance of time among product versions	1. Operator skill, motivation
2. Cycle time	2. Tools required
3. No of stations	3. Tools change necessary
4. Traffic problems	4. Setup time
5. Station space	5. Buffer allocation
6. Transportation networks	6. Average station time
7. Communication among the groups	7. Variance of time among product versions (diff. models)
8. Task complexity	8. Ergonomic values (required grip strength)
9. Reliability	9. Need of storage
	10. Working place
	11. Worker absenteeism during operation

Table 1. Performance and workstation indexes for ALB and sequencing

AL design model and solution methodology combine the model stage. Design tools are modelled and formulated after collection and verification of the input data. Design tools modelling include the output data, interaction between different modules and methods to be developed. Wide range of heuristic as Branch and Bound search, Positional weight method, Kilbridge and Wester Heuristic, Moodie-Young Method, Immediate Update First-Fit (IUFF), Hoffman Precedence Matrix (Ponnambalam et al., 1999) and meta-heuristic based solution strategies as Genetic Algorithm GA (Sabuncuoglu et al., 1998), Tabu Search TS (Chiang, 1998) , Ant Colony Optimization ACO (Vilarinho & Simaria, 2006), Simulated Annealing SA (Suresh & Sahu, 1994) for ALB problems are adopted in industrial and research level (figure 2). Validation of the models is a result of performance towards the objectives of that particular line.
Line performances of AL design is a measure of multi-objective characteristics. Varied objective functions are considered for ALB (Tasan & Tunali, 2006). Designer's goal is to design a line considering higher efficiency, less balance delay, smooth production, optimized processing time, cost effectiveness, overall labour efficiency and just in time production (JIT). The aim is to propose a line by exploiting the best of the design methods which will deal in actual fact with user preferences.

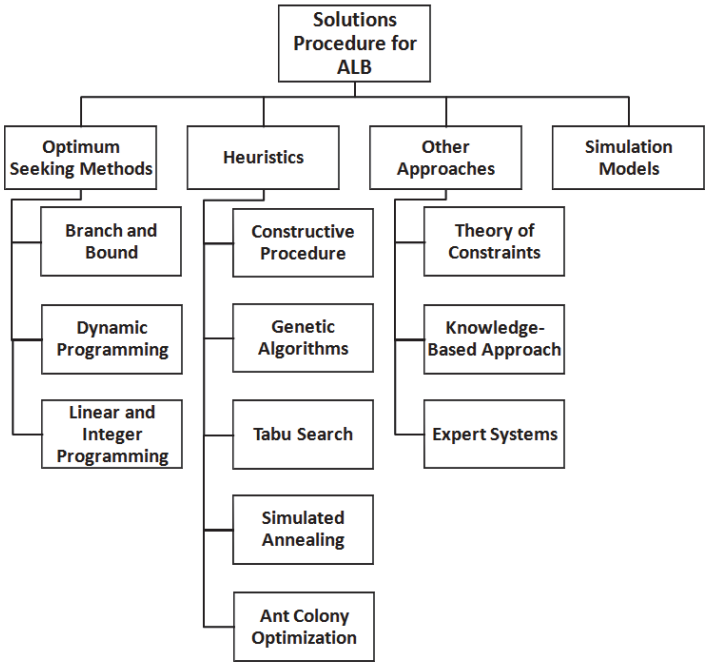

Fig. 2. Different solution procedures for ALB

Design evaluation refers to a user friendly developed interface where all necessary AL data is accessible extracted from different database. Validation of different algorithms and methods is integrated with different design packages (Rekiek & Delchambre, 2006).

2.1 Classification of ALB problems

Classification of ALB problem is primarily based on objective functions and problem structure. Different versions of ALB problems are introduced due to the variation of objectives (figure 3).

Objective Function Dependent Problems:

- Type F: Objective dependent problem, it is associated with the feasibility of line balance for a given combination of number of stations and cycle time (time elapsed between two consecutive products at the end of the AL).
- Type 1: This type of problem deals with minimizing number of stations, where cycle time is known.
- Type 2: Reverse problem of type 1.
- Type E: This type of problem is considered as the most general version of ALBP. It is associated with maximizing line efficiency by minimizing both cycle time and number of stations.
- Type 3, 4 and 5: These corresponds to maximization of workload smoothness, maximization of work relatedness and multiple objectives with type 3 and 4 respectively.

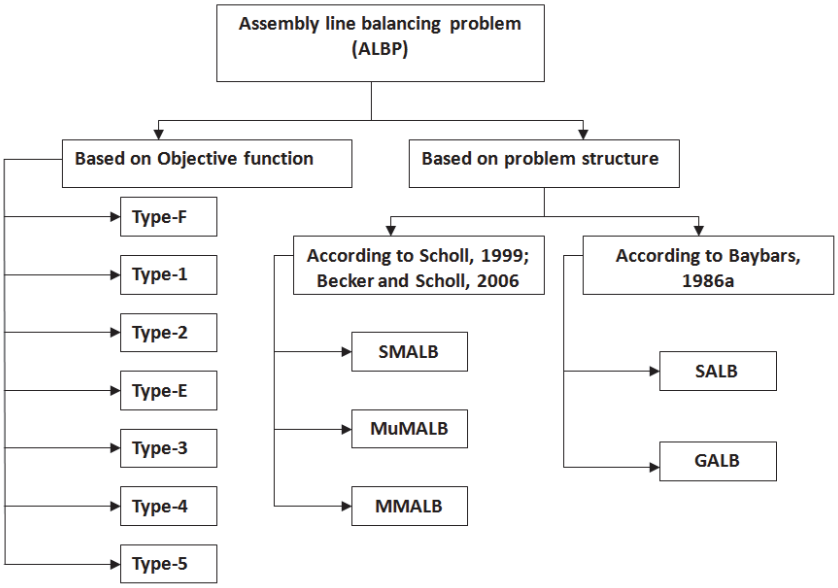

Fig. 3. Classification of ALBP (Scholl & Boyesen, 2006)

Problem Structure Dependent Problems:

- SMALB: This refers to single model ALB problems, where only one product is produced.
- MuMALBP: Multi model ALB problems, where more than one product is produced on the line in batches.
- MMALBP: Mixed model ALB problems, various models of a generic product are produced on the line in an intermixed situation.
- SALBP: Simple ALB balancing problems, simplest version of balancing problems, where the objective is to minimize the cycle time for a fixed number of workstation and vice versa.
- GALBP: A general ALB problem includes those problems which are not included in SALBP. Those are for instance, mixed model line balancing, parallel stations, U-shaped and two sided lines with stochastic task times.

2.2 MMALB and sequencing

Production system planning usually starts with the product design initially. The reason behind this, a great deal of future costs is determined in this phase. Initial configuration and installation of productive units triggers the actual cost of the production planning of assembly system. Resources required by the production process also determines by the configuration planning. Different methodologies are utilized as depicted in figure 2, to support the configuration planning which are included under the term ALB.

In the case of mixed model lines, different models often utilize available capacity in very different intensities. Therefore modification of balancing or line re-balance might be necessary. A family of products is a set of distinguished products (variants), whose main

functions are preferably similar, usually produced by mixed-model lines. Mixed model lines are generally employed in the cases (Rekiek & Delchambre, 2006), where

- The cycle time is usually greater than a minute.
- The line price cannot be amortized by a single product model.
- The product must not be delivered in a short time.
- Each product is quite similar to others.
- The same resources are required to assemble the products.
- The set up time of the line needs to be short.

MMALBP occurs when designing or redesigning a mixed-model assembly line. This is subjected to find a feasible assignment of tasks to workstations in such a manner that production demand of different product variants are met within the defined shift times. Minimization of assembly costs, satisfying the constraints are also a concern. Mixed model lines are classified into two different types, which are referred as dual problems.

- MMALBP-1: minimizes the number of workstation for a given cycle time.
- MMALBP-2: Minimizes the cycle time for a given number of workstation.

In type 1 problem cycle time or, the production rate, is pre-specified. That is why; it is more frequently used to design a new AL where demands are forecasted beforehand. Type 2 problem deals with maximization of production rate of an existing AL. This is applied for example when changes in assembly process or in product range require the line to be redesigned.

Mixed model sequencing aims to minimize sequence dependent work overload. Sequencing is based on a detailed model scheduling depending on the operation times, worker movements, necessary tool changes required, station borders and other characteristics of the line. Different models are composed of different product options and thus require different materials and parts, so that the model sequence influences progression of material demand over time. As ALs are commonly coupled with preceding production levels by means of a just in time (JIT) supply of required materials, the model sequence need to facilitate this. An important prerequisite for JIT-supply is the steady demand rate of materials over time, as otherwise advantages of JIT are sapped by enlarged safety stocks that become necessary to avoid stock outs during the peak demand. Accordingly, JIT centric sequencing approaches aim at distributing the material requirements over the planning horizon (Boyesen et al., 2007).

3. Methodology for solving MMALB and sequencing

A logic and mathematical formulation based methodology is proposed for solving MMALB and sequencing. During the development of this approach, a constant speed AL is considered where task transportation, machine setup and tool changing times are taken within the task times. Task time of each model, precedence relations of tasks are known whereas work in progress buffers, station parallelization, assignment restrictions, zoning constraints are not allowed. MMALB problem type 2 is considered. The balancing is achieved in two consecutive stages which are named as first stage and second stage.

3.1 First stage: balancing from equivalent single model

Balancing in this stage finds locally optimized solution in the first stage iteration. Objective of this stage is to find solution(s) with specified number of stations with a minimum cycle time. Solutions are considered as locally optimized as the principle objective is to find a solution which will define a smooth production by minimizing objective function of second

stage. The concept of ALBP-1, where the aim is to optimize the number of workstations with a predefined fixed cycle time is utilized in first stage of this proposed approach. The fixed cycle time is considered as the solution lower bound, CT_{min} for finding desired station numbers, CT_{min} is increased by 1 sec per iteration. Solution lower bound is determined with minimum cycle time (Gu et al., 2007) as:

$$CT_{min} = max\left[\frac{1}{S}\sum_{i=1}^{n} t_i, max\, t_i\right] \quad (1)$$

Where, t_i is the i_{th} task time and S is the desired number of stations. The flow diagram of first stage is illustrated in figure 4.

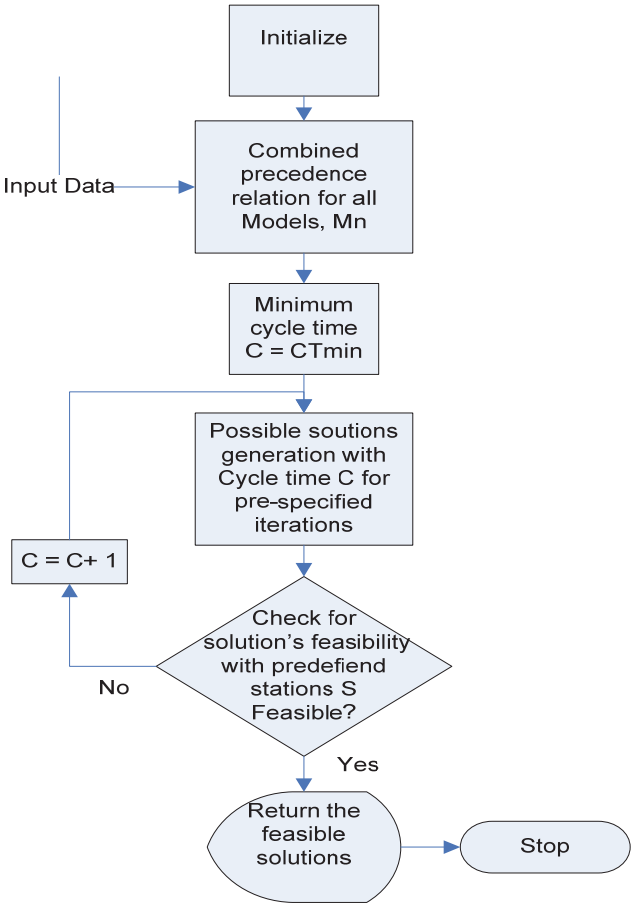

Fig. 4. Flow diagram of first stage iteration

Tasks of different models are first considered as an equivalent single model. Combined precedence diagram alter different models into one equivalent single model. A simple combined precedence relation example is given in figure 5, with 12 tasks, where node containing the task number and the values indicate tasks time.

The following algorithm defined as step by step procedure, generates a number of feasible solutions for equivalent single model. Optimized feasible solutions are stored as the input solutions of second stage.

1. Open a new station S_1 with a cycle time $C = CTmin$.
2. Determine the set of tasks without predecessor, $s = \{i, j \ldots . . n\}$
3. Assign randomly one task from s in station S_1.
4. Remove the assigned task from the precedence graph, update station time as the cycle time $C = CTmin - t_i$
5. Update set of tasks without predecessor as $s = \{j, k \ldots . n\}$
6. Assign tasks randomly from s to S_1 until C is positive and update C and s each time.
7. When C is negative or zero for randomly assigned any task from s, check for the other tasks in s to be fitted in S_1.
8. When C is negative or zero for all the tasks in existing s, open a new station S_2 and $C = CTmin$ is restored for S_2.
9. Repeat steps 1 to 8 until the assignment of all tasks.
10. Generate all feasible solutions.
11. Check the solutions with predefined station numbers. If the solutions are not feasible, repeat the above steps with $C = CTmin + 1$ and so on until the desired number of stations are met.
12. When a number of feasible solutions are achieved, store finally updated C as the cycle time. Store and return the workstation based solutions with the station assignment information for next stage.

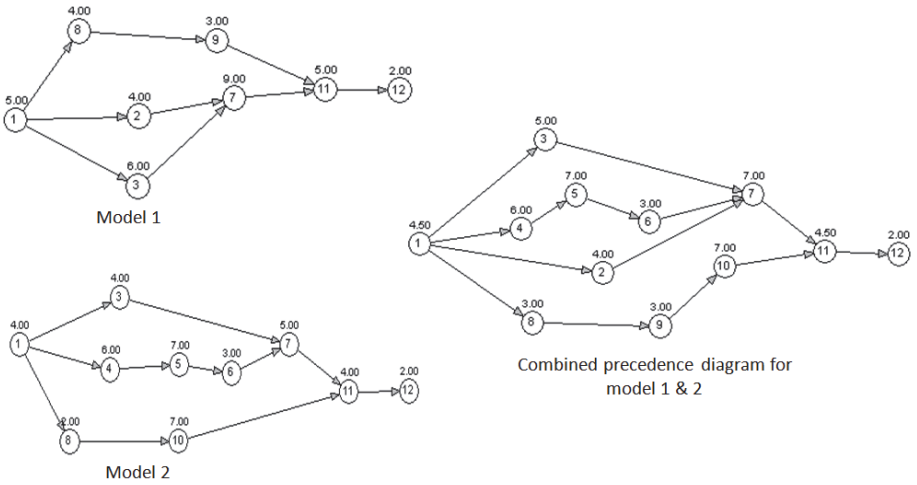

Fig. 5. Combined Precedence diagram for model 1 and 2

3.1.1 First stage experimentation

Benchmarked 'Buxey' data sets of 29 tasks for SMALBP-2 (Scholl, 1993) are tested with first stage balancing approach. Precedence matrix (table 2) defines the precedence constraints associated to the tasks. Precedence task set 1, 2 refers task 2 precedes task 1 in a $\{0, 1\}$ task matrix where column precedes the row. A 1 is placed where there is a precedence relation, otherwise 0. Solution flexibility can be determined from precedence matrix by

measuring $F-ratio$ (flexibility ratio). Higher $F-ratio$ indicates less precedence constraints and greater flexibility in generating multiple feasible solutions (Rubinovitz et al., 1995).

$$F-ratio=\frac{2\times Z}{K(K-1)} \quad (2)$$

Where, Z is the number of zeros above the diagonal and K is the number of task elements. $F-Ratio$ for the combined precedence diagram of figure 5 is 0.78.

Tasks	1	2	3	4	5	6	7	8	9	10	11	12
1		1	1	1	0	0	0	1	0	0	0	0
2	0		0	0	0	0	1	0	0	0	0	0
3	0	0	D	0	0	0	1	0	0	0	0	0
4	0	0	0	I	1	0	0	0	0	0	0	0
5	0	0	0	0	A	1	0	0	0	0	0	0
6	0	0	0	0	0	G	1	0	0	0	0	0
7	0	0	0	0	0	0	O	0	0	0	1	0
8	0	0	0	0	0	0	0	N	1	0	0	0
9	0	0	0	0	0	0	0	0	A	1	0	0
10	0	0	0	0	0	0	0	0	0	L	1	0
11	0	0	0	0	0	0	0	0	0	0		1
12	0	0	0	0	0	0	0	0	0	0	0	

Table 2. Precedence matrix for combined precedence diagram for figure 5

First stage MATLAB program compiled for 'Buxey 29 tasks Problem' (Scholl, 1993) and the task times are shown in table 3.

Task No.	Time, Sec	Task No.	Time, Sec	Task No.	Time, Sec
1	7	11	21	21	1
2	19	12	10	22	9
3	15	13	9	23	25
4	5	14	4	24	14
5	12	15	14	25	14
6	10	16	7	26	2
7	8	17	14	27	10
8	16	18	17	28	7
9	2	19	10	29	20
10	6	20	16	-	-

Table 3. Task times of 'Buxey' benchmarked problem

3.1.2 Experiment results

First stage generates multiple feasible solutions for different number of stations. Tasks assignment is shown below, where S1 to S9 represents predefined 9 stations with assigned tasks. Minimum cycle time achieved 37 seconds which fulfil the benchmarked solution result. Station assignments of the tasks are: S1 {2, 7, 9, 10, 26}, S2 {1, 6, 12, 27}, S3 {3, 4, 5, 14}, S4 {8, 11}, S5 {13, 17, 25}, S6 {15, 16, 20}, S7 {18, 19, 21, 22}, S8 {23, 28}, S9 {24, 29}.

'Buxey' 29 tasks problem			
Benchmarked Results		Stage1 procedure	
Predefined stations, m	*Minimal cycle time*	*Minimal cycle time C*	*CPU run time, sec*
7	47	48	193.83
8	41	41	136.04
9	37	37	105.45
10	34	34	85.45
11	32	32	73.46
12	28	30	50.82
13	27	27	24.42
14	25	25	8.91

Table 4. Comparison between benchmark results and stage1 procedure

Benchmark results and the results obtained by first stage balancing are depicted in table 4. Figure 6 shows line balancing solution for 'Buxey' 9 station problem obtained by first stage balancing procedure.

Fig. 6. Workstations Vs Workload (37 sec cycle time)

3.2 Second stage: balancing for mixed-models

This stage finds optimal solutions for mixed-models with the results achieved from first stage. Feasible solutions generated from the first stage are decoded and scaled with second stage objective function. The aim is to obtain the best solutions from first stage in terms of second stage objective which ensures a minimal balance delay. The feasible solutions of first stage are coded as the workstation based solutions. Workstation based solution representation scheme is shown in figure 7.

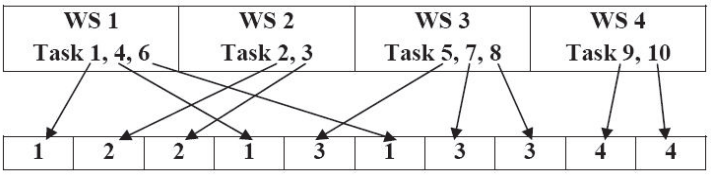

Fig. 7. Workstation based solution representation

Inputs for second stage objective function from the generated first stage solutions are as follows:

1. Number of workstations S, represented by the solution which is the highest numerical number of the solution.
2. No of tasks K in precedence graph as the length of the solution.
3. Tasks assignment in workstations according to the solution representation scheme.
4. The initial problem definition of MMALB-2 describes the inputs to the objective function are number of models to be produced M, production demand for each model N_m, where $m = 1\ to\ M$ and task times for each model t_{km}.

3.2.1 Objective function formulation

Objective function considered for MMALBP-2 to facilitate a smooth workload balance among the stations, while taking smoothed station assignment load into consideration. It also optimizes shift time as cycle time of single model case is replaced by shift time in mixed-model balancing.

Notations:

M	Number of models to be produced.
N_m	Scheduled quantity to be produced for each model where $m = 1$ to M.
T	Shift time period for the scheduled quantity to be produced.
K	Number of total tasks.
CT_{min}	Minimum cycle time.
t_{km}	Task times where $k = 1$ to K and m = 1 to M. t_{km} represents the work time of task k on model m.
E_k	Total time required to complete $\sum_{m=1}^{M} N_m$ units in the scheduled period for task k
S	Number of stations.
Q_{sm}	Amount of time that the operator in station s is assigned on each unit of model m
T_s	Station time where $s = 1$ to S.
P_{sm}	Total time assigned to station s on model m.
P_m	Average amount of total work content for all units of model m assigned to each station.

All models of production demand can be summarized as the total products to be produced, where;

$$\text{Total products to be produced} = \sum_{m=1}^{M} N_m \tag{3}$$

The total task times required to complete the production of all models are:

$$E_k = \sum_{m=1}^{M} N_m \times t_{km} \tag{4}$$

In MMAL, operation time is denoted as Q_{sm}; where $s = 1\ to\ S$ and $m = 1\ to\ M$; which refers the amount of time required in station s for each unit of model m. Mixed-model line balancing solutions are obtained here from the single model balancing algorithm of first stage by replacing cycle time C to shift time period T. Total time assigned to station s in period T can be defined as

$$T_s = \sum_{m=1}^{M} N_m \times Q_{sm} \tag{5}$$

Total time assigned to station s on model m in period T is

$$P_{sm} = N_m \times Q_{sm} \tag{6}$$

Now, P_m represents average or desired amount from the total work content for all units of model m assigned to each station and P_m can be presented as

$$P_m = N_m \times \frac{\sum_{k=1}^{n} t_{km}}{S} \tag{7}$$

Hence, minimizing the value of $(P_m - P_{sm})$ points to smooth out or equalize total work load for each model over all work stations. Therefore the objective function Y $(SSAL, Smoothed\ Station\ Assignment\ Load)$, can be abridged as to minimize the following function

$$Y = \min \sum_{s=1}^{S} \sum_{m=1}^{M} [P_m - P_{sm}] \tag{8}$$

3.2.2 Mixed-model line sequencing

Tasks associated to ALs are mostly dealing with the repetitive periodic tasks occurring at a regular interval. A static AL's task sequencing heuristic (Askin & Standridge, 1993) is integrated to the results of MMALB-2 obtained from second stage. The objective of sequencing is to generate a dispatch system which controls the order of entry of the products to the first station.

Let, q_m is the proportion of product type m to be assembled in the line where $m = 1\ to\ M$. The initial step is to develop an AL balance for the weighted average product. Let t_{km} is the task time for k of model m and S_s is the set of tasks assigned to station s where $s = 1\ to\ S$. In that case if the cycle time is CT, the average feasibility condition can be stated as:

$$\sum_{k \in S_s} \sum_{m=1}^{M} q_m t_{km} \leq CT \tag{9}$$

This condition indicates the averaged across all items produced in the long term, no workstation is overloaded. According to the feasibility condition, one single product ALB problem needs to be solved. Due to this, task time of task k can be summarized as:

$$t_k = \sum_{m=1}^{M} q_m t_{km} \tag{10}$$

For each model m, Q_m amount need to be produced. If r be the greatest common denominator of all Q_m a repeating cycle comprised of $N_m = Q_m/r$ units should suffice where the models are from $m = 1\ to\ M$. The cycle would be repeated r times to satisfy the period demand. In that case, $N = \sum_{m=1}^{M} N_m$ items are produced in each cycle.

The objective of designing such cycle is to define a smooth production rate of each model type. This will also prevent the excessive idle time at the workstation due to the mix-induced starving of workstations. A workstation is starved if on completion of all the defined tasks, there are no tasks available for it to work on because the next task has not been completed in the prior station. This is even more crucial in the bottleneck stations. That is why, the maintaining of a smooth flow of the parts to those stations is necessarily important. The bottleneck stations are the stations with maximal total work or equivalently average work load per cycle. If a partial sequence overloads this workstation with respect to average cycle time CT, the subsequent stations are starved. If a partial sequence under loads the bottleneck station, the initial output rate from the line will be too high which will result in accumulating the inventory. In case of the relative workload of station s is C_s, it workload can be defined as:

$$C_s = \sum_{k \in S_s} t_k \tag{11}$$

The bottleneck station S_b is the station with maximum workload or equivalently or average workload per cycle. Hence,

$$S^b = \text{argmax}_s \, C_s \tag{12}$$

Let, X_{mn} is the value equals to 1 if model m is placed in the n_{th} position and 0 otherwise. In that case, $m(n)$ will indicate the type of model placed in n_{th} position in the assembly sequence. Now, the approach is to select the n_{th} model to be started to insert in the line to optimize as following:

$$\text{minimize maximum}_{1 \le n \le N} \left[\sum_{m=1}^{M} \sum_{k \in S_{s^b}} t_{km(n)} - nC_{s^b} \right] \tag{13}$$

Sequencing is done in two consecutive steps:
Step 1: Initialization, create a list of all products to be assigned during the cycle and named as list A.
Step 2: Assign a product. For $n = 1 \; to \; N$ from list A, create a list B of all product types that could be assigned without violating any constraints. From list B select the product type m' that minimizes the objective function of equation 13. Add model type m' to the n_{th} position. Remove a product type m' from list A and if $n < N$, go to step 2. C_{s^b} defines the time accumulated in bottleneck stations.
Aim of this sequencing heuristic is to create a list of unassigned products first, which is then reduced first to a list of feasible assignable products and to the single best feasible products. The assumption of this heuristic is that the operator in manual workstations can intermix to a slight degree to keep the line moving even if the station is temporarily overloaded.

4. Case study

A modified practical problem definition of WXYZ Company (Askin and Standridge, 1993) is considered here for the implementation of the addressed integrated approach for MMALB-2 and sequencing. The problem defines assembly of web cameras of four different models. A constant speed, conveyor based, straight AL is considered where tasks contains no zoning constrains, capacity constraints or assignment restrictions. Average demands per shift for four different types of cameras are 20 units of model 1, 30 units of model 2, 40 units of model 3 and 10 units of model 4. Aim is to balance the line for mixed-model assembly system with optimized shift time. Assembly module has four fixed workstations (MMALB-2) where they have decided to place one operator in each station. Each workstation is capable of performing the same set of operations on all four model types. Task times (sec) for each model are shown in table 5.
Now, following the proposed methodology, the aim is to find:

- Optimal cycle time accounting for workstation availability considering combined task relationships for all models (first stage).
- Distributing the tasks of all four models to four different workstations maintaining an overall workload balance, *i.e. SSAL* as the objective of mixed-model balancing considered here and also to find out optimized overall shift timing for assembly of all models according to demand (second stage).
- Finally, constructing a repetitive lot planning through model sequencing (mixed-model sequencing).

Tasks	Model M1	Model M2	Model M3	Model M4	Wt. Avg.	Immediate predecessors
Op 1	14	34	15	10	19	-
Op2	12	15	11	17	14	Op 1
Op 3	39	47	40	51	45	Op2
Op 4	3	4	4	7	5	Op 1
Op 5	11	13	10	9	11	Op 3
Op 6	19	29	21	21	23	Op 4
Op 7	11	14	9	10	11	Op 5
Op 8	21	38	28	32	30	Op 3, Op 6
Op 9	13	19	15	17	16	Op 5, Op 8
Op 10	33	41	42	43	40	Op 7, Op 9
Total	176	254	195	216	234	-

Table 5. Tasks time per model

Ten different tasks or operations are identified for completing the assembly of each model. Task times are different depending on the models. Combined precedence diagram for four models are shown in figure 8.

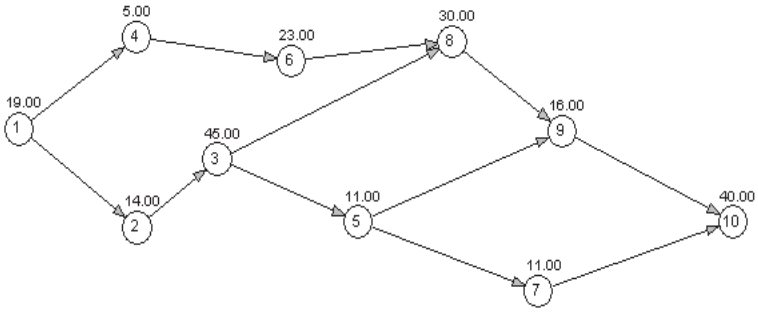

Fig. 8. Precedence diagram of the case problem

Proposed first stage generates two feasible solutions considering minimum cycle time for the case problem. Cycle times of both workstation based solutions are 59 seconds. Next step is to decode and scale the optimized solutions to achieve the best one considering overall SSAL. Feasible solutions represented in figure 9, decoded in table 6, 7.

1	2	2	1	3	1	3	3	4	4

1	2	2	1	3	1	4	3	3	4

Fig. 9. Feasible solutions of the case problem

Work Station	Assigned Tasks	Station Time
1	Op 1, Op 4, Op6	47
2	Op2, Op3	59
3	Op5, Op7, Op8	52
4	Op9, Op10	56

Table 6. Decoded first solution from figure 9

Work Station	Assigned Tasks	Station Time
1	Op 1, Op 4, Op6	47
2	Op2, Op3	59
3	Op5, Op8, Op9	57
4	Op7, Op10	51

Table 7. Decoded second solution from figure 9

Two feasible solutions are explored and scaled with the objective function of second stage. Results obtained are illustrated in table 8. Overall SSAL are 22 and 23.9 for solution 1 and 2. Therefore solution 1 has the better smoothed stations assignment load.

Feasible Solutions	Stations	St. Time(Hr) per shift for mixed-models	SSAL (Y value of the objective function)
Solution 1	1	1.31	7.85
	2	1.56	4.15
	3	1.44	1.65
	4	1.54	5.35
Solution 2	1	1.31	7.85
	2	1.55	4.15
	3	1.58	4.25
	4	1.42	3.55

Table 8. Shift times and SSAL values for generated solutions of figure 9

Production ratio of four models is 2:3:4:1 according to demand. Therefore, a repetitive lot of 10 units need to be considered. As a consequence of demand fluctuation, the ratio may vary but the goal is to find feasibility of a long run path with demand ratio (Askin and Standridge, 1993). The feasible solution of the mixed-model balancing indicates station 2 as bottleneck station as the cycle time of 59 sec is fully consumed. Bottleneck station load per model are 51, 62, 51 and 68 seconds.

According to this sequencing heuristic, initially all models are eligible since the cumulative production level deficit is below one for all models. The sequencing is shown in table 9. M2 is selected to minimize the maximum deviation of actual to desired production for any assignable product. If the presence of bottleneck stations are multiple, larger of the deviation are chosen for constructing the model sequencing. Selection of M2 puts the production schedule $[1 - 0.3]$ or 0.7 ahead of the schedule for M2 and 0.2, 0.4 and 0.1 behind for M1,

M3 and M4. In second step ($n = 2$), selection of M2 is not eligible because its assignment will place M2 $[1 - (-0.4)]$ or 1.4 ahead of the schedule. Following this heuristic, a recurring lot planning of 10 units where 2 units of model 1, 3 units of model 2, 4 units of model 3 and 1 unit of model 4 are achieved for the case problem where the sequence of mixed-models would be M2-M3-M4-M1-M3-M2-M3-M2-M1-M3 with a shift time of 1.56 hours.

Step	Models If Selected				Selected Model	WS2 Load (Bottleneck)
1	M1	M2	M3	M4	M2	62(3)
	8,0.2	3, 0.3	8, 0.4	9, 0.1		
2	5, 0.4	6, -0.4	5, 0.8	12, 0.2	M3	113(-5)
3	13, 0.6	2, -0.1	13, 0.2	4, 0.2	M4	181(4)
4	4, 0.8	7, 0.2	4, 0.6	-	M1	232(-4)
5	8,0	3,0.5	8,1	-	M3	287(-8)
6	16, 0.2	5,0.8	16,0.4	-	M2	359(5)
7	13, 0.4	2,0.1	13, 0.8	-	M3	400(-13)
8	21, 0.6	10,0.4	21,0.2,	-	M2	482(10)
9	2, 0.8	-	2, 0.6	-	M1	529(-2)
10	-	-	0,1	-	M3	590(0)

Table 9. Mixed-model sequencing for the case problem

Most solutions for ALB problems look for one final optimized solution. However, it is fairly important to explore the alternative solutions (Boysen, 2006). This integrated approach facilitates such necessary diversity of the solutions. If 8 station 'Buxey' data sets are focused, three feasible solutions are generated with 41 seconds minimal cycle time. As in the case of mixed-model balancing, the objective function is measured from the solutions obtained by the joint precedence graph, feasible solutions need to satisfy the performance indexes of the line. Such performance indexes are the line efficiency, station smoothness index and the overall balance delay. Generated workstation based solutions are depicted in figure 10.

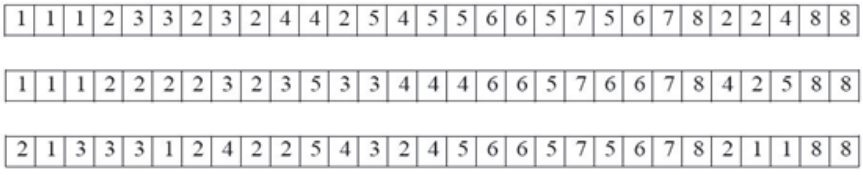

Fig. 10. Generated 3 feasible Solutions with the first stage approach for 8 Station 'Buxey' problems

As a consequence of the generated balancing solutions, corresponding station load and utilization of the stations for three solutions are depicted in figure 11.

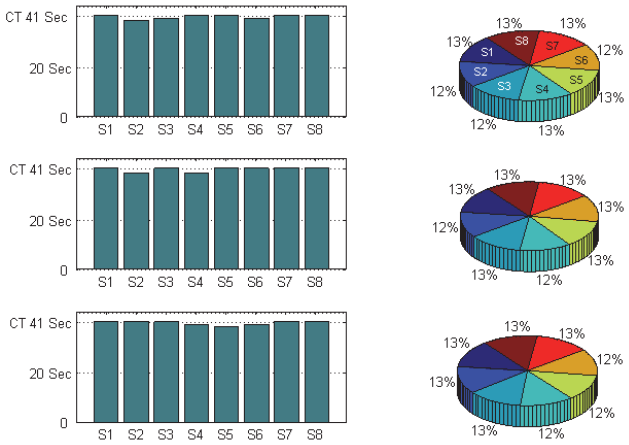

Fig. 11. Station Load over 41 sec Cycle time and consecutive station utilization% for 3 solutions

Line efficiency *LE* gives an impact of percentage of utilization of the line (Boysen et al., 2006). *LE* for generated 3 feasible solutions is 98.2 % as most of the stations are fully utilized with equivalent single model case. Smoothness index (SX) is measured to indicate the relative smoothness of the AL balance (Ponnambalam, et al., 1999). A smoother line results in reducing process inventory as well as smoothed workload balance. SX for all the generated 3 solutions is 2, which indicates the solutions are feasible and having less balance delay.

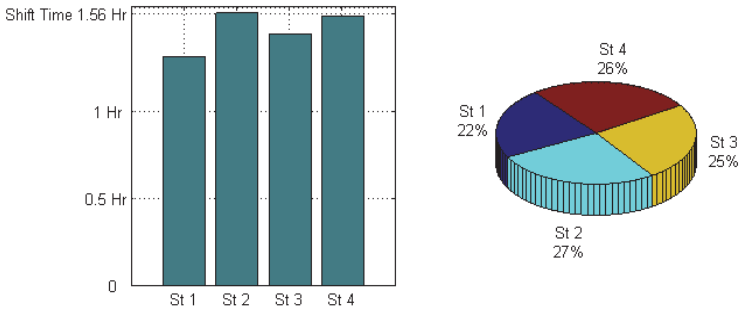

Fig. 12. Shift timing and station utilization of the case problem of MMALB-2

In mixed-model balancing, work elements are assigned to different workstations on a daily basis or an entire shift instead of cycle time basis as is done in single model case. The objective function considered here is to distribute evenly the total entire workloads within the shift time. As in example case problem, selected optimal solution for mixed-model case is solution 1. Station per SSAL values are 7.85, 4.15, 1.65, 5.35 and overall SSAL for the solution is 22. The solution having a smallest value of SSAL indicates optimality of workload balance among the stations. For assembling the entire 100 units of 4 different

models, generated optimal solution indicates shift timing of 1.56 hours. Station 2 is fully utilized where as station 1 having the idlest time during an entire shift. Shift timing and station utilization are illustrated in figure 12.

5. Conclusions and future works

Systematic design and balancing of ALs is somewhat complicated, especially for the large scale product customization due to the uneven nature of tasks times of different models. This is the parallel rationale of having a difficulty to a smooth workload balance among workstations. But, in terms of not finding a good balancing structure supported by a proper sequencing of the mixed models, the interim performances of the line will be poor which obstruct the overall mixed-model AL-based production scenario.

The research carried out in this manuscript helps to identify the critical design parameters associated to ALB and sequencing. It also briefly addresses the overall problem domain of ALB and sequencing. The methodology for MMALB and sequencing addressed in this work distributes workloads of mixed-models to predefined workstations considering smoothed station assignment load. This results in optimizing the shift timing of AL for any combination of various models and defines a repetitive production lot planning from model sequencing. The end result can be summarized as maximization of production rate.

It can be concluded from the experimental results that the addressed two stage balancing and sequencing methodology ensures a smooth and optimal production with varied demand for MMALB-2 in ideal conditions. Whereas, the first stage procedure can also be implemented for Single model ALB problems. Proposed approach is shown to perform well as the optimized solution generation scheme is converged from the different feasible solutions. Integrated sequencing approach of this work also imparts an understanding of a smooth production of the mixed-models by defining a repetitive production schedule. Overall, the results of this work are important when designing and balancing an AL layout from the scratch or redesigning for product customization.

In a more complex assembly environment, there might be several constrains like equipment restrictions, facility layout restrictions, buffer allocation and stations length which essentially differ from plant to plant. For an overall understating of extensive performances of MMALB and sequencing, all those plant and line oriented constraints should be taken into account within the balancing methodology and this is considered to be the future extension of this work.

6. Appendix: MATLAB codes for the case study

Weighted task times representation: Cost Function

```
function c= cost()
c = [19 14 45 5 11 23 11 30 16 40];
```

Precedence matrix Representation: Graph Function

```
function g= graph()
g = [0 1 0 1 0 0 0 0 0 0
     0 0 1 0 0 0 0 0 0 0
     0 0 0 0 1 0 0 1 0 0
     0 0 0 0 0 1 0 0 0 0
     0 0 0 0 0 0 1 0 1 0
```

```
0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0];
```

The proposed approach for assigning the tasks:

```
function  [stInfo loadInfo stationCount ] = assign(tTime, nOfTask, Ctmin, tMatrix);
   global readyTaskList;
   global readyTaskCount;
   global taskCount;
   global parentCount;
   global taskMatrix;
   global taskTime;
   global cTime;
   global stCount;
   global stID;
   global loadID;
   global minStationID;
      stCount = 0;
   cTime  = Ctmin;
   taskTime = tTime;
   taskCount = nOfTask;
   taskMatrix = tMatrix;
   parentCount = buildParentCount();
   readyTaskCount = 0;
   buildReadyTaskList();
   minStationID = ones( 1, taskCount);
      while readyTaskCount > 0.5
      t  = getReadyTask();
      loadTask(t);
      updateReadyTaskList(t);
   end
      stInfo = stID;
   loadInfo = loadID;
   stationCount = stCount;
   function  loadTask(t);
   global taskTime;
   global stTimeLoad;
   global stTaskLoad;
   global stID;
   global loadID;
   global minStationID;
      reqTime = taskTime(t);
   minSID = minStationID(t);
   st = getStation( reqTime, minSID);
   stTaskLoad( st) = stTaskLoad( st) + 1;
   stTimeLoad(st) = stTimeLoad(st) + reqTime ;
   stID(t) = st;
```

```
   loadID(t) = stTaskLoad( st) ;
function st = getStation( reqTime, mnstID);
   global stCount;
   global stTimeLoad;
   global stTaskLoad;
   global cTime;
   st = -1;
    for i= 1:stCount
      if stTimeLoad(i) +  reqTime < cTime + 0.5 && mnstID < i + 0.5
         if st > 0
            if stTimeLoad(i) > stTimeLoad(st)
               st = i;
            end
         else
             st = i;
         end
      end
   end
    if st < 0
       stCount = stCount +1;
       stTimeLoad(stCount) = 0;
       stTaskLoad(stCount) = 0 ;
       st = stCount;
   end
function  pCount = buildParentCount();
   global taskMatrix
   pCount = sum(taskMatrix, 1);
function buildReadyTaskList();
   global parentCount;
   global taskCount;
   for t = 1:taskCount;
      if parentCount(t) == 0
         addReadyTask(t);
      end
   end
function  updateReadyTaskList(t);
    global parentCount;
   global readyTaskList;
   global readyTaskCount;
   global taskCount;
   global taskMatrix;
   global minStationID;
   global stID;
    for i = 1:taskCount;
      if taskMatrix(t,i) > 0.5 % if dependency exist
         parentCount(i) = parentCount(i)-1;
          if minStationID(i) < stID(t)
            minStationID(i) = stID(t);
         end
```

```
        if parentCount(i) < 0.5 % if no parent exist
          addReadyTask(i);
        end
      end
    end
function  rTask  = getReadyTask();
    global readyTaskList;
    global readyTaskCount;
    ind = getRandIndex(readyTaskCount);
    rTask = readyTaskList( ind );
    readyTaskList( ind ) =  readyTaskList( readyTaskCount );
    readyTaskCount = readyTaskCount -1;
function addReadyTask(t);
    global readyTaskList;
    global readyTaskCount;
    readyTaskCount = readyTaskCount + 1 ;
    readyTaskList( readyTaskCount ) = t;
function  ind = getRandIndex(readyTaskCount);
    ind = rand;
    ind = ind * readyTaskCount;
    ind = round(ind);
    if ind < readyTaskCount;
       ind = ind +1;
    end
```

Solution Generation: Schedule Generator

```
close all;
clear all;
stCount = inf;
maxStation = 4
G_RAPH =graph();
C_OST = cost();
 [temp, taskCount] = size(C_OST);
%ctMin = max(C_OST);
TTM = [14 12 39 3 11 19 11 21 13 33 % task times t1:t10 for model 1
      34 15 47 4 13 29 14 38 19 41 % task times t1:t10 for model 2
      15 11 40 4 10 21 9 28 15 42 % task times t1:t10 for model 3
      10 17 51 6 9 21 10 32 17 43]; % task times t1:t10 for model 4
 [nom,not] = size (TTM) % no of models, no of tasks
%NOS = 4 % predefined number of stations
% weighted average
tw = ceil(sum(TTM(1:nom,1:not))/4)
max_tw = max(max(tw)) % maximum task time
cmin = floor(sum (tw)/maxStation)
ctMin = max(cmin, max_tw) % minimum cycle time, lower bound
%C = CTmin
 solutionCount = 0;
while(stCount > maxStation)
      for i=1:300 % no of iteration
```

```
    [stInfo ldInfo stationCount ] = assign(C_OST, taskCount, ctMin, G_RAPH);
    if stCount > stationCount
       stCount = stationCount;
    end
    if stationCount <= maxStation
       flag = 0;
       j =1;
       while j<=solutionCount &&  flag<0.5
          %v1 =   sum(abs(loadInfo(j,1:taskCount) - ldInfo(1:taskCount) ));
          v2 =   sum(abs(sationInfo(j,1:taskCount) - stInfo(1:taskCount) ));
          if  v2 < 0.5 % v1 <0.5 &&
             flag = 1;
          end
          j = j+1;
       end
       if flag < 0.5
          solutionCount = solutionCount + 1;
          sationInfo(solutionCount,:) = stInfo(1:taskCount);
          loadInfo(solutionCount,:) = ldInfo(1:taskCount);
          NumberOfStation(solutionCount) = stationCount;
       end
    end
  end
  if  stCount > maxStation
     ctMin = ctMin +1;
  end
end
sationInfo
loadInfo
Updated_Final_CYCLE_TIME = ctMin
```

Mixed Model Scaling for optimized SSAL value:

```
s1 = [1 4 6] % Task Assigned to Station 1
s2 = [2 3]
s3 = [5 7 8]
s4 = [9 10]
time_s1 = sum(tw(s1))
time_s2 = sum(tw(s2))
time_s3 = sum(tw(s3))
time_s4 = sum(tw(s4))
st = [s1 s2 s3 s4];
Nm= [20 30 40 10] % demand for each model
E = (Nm*TTM); % total time required to complete all Nm (100) units
%T = sum(E(st(1:end)))
T1 = sum(E(s1(1:end))) %Station 1 work load for all models
T2 = sum(E(s2(1:end))) %Station 2 work load for all models
T3 = sum(E(s3(1:end))) %Station 3 work load for all models
T4 = sum(E(s4(1:end))) %Station 4 work load for all models
ST_TIME = [T1 T2 T3 T4]/3600 % In hour
 Station_time= [T1 T2 T3 T4]
```

```
L1 = TTM(1,:);
L2 = TTM (2,:);
L3 = TTM(3,:);
L4 = TTM(4,:)
 W = [sum(L1) sum(L2) sum(L3) sum(L4)];
 Qsm1 = [(sum(L1(s1(1:end)))) (sum(L2(s1(1:end)))) (sum(L3(s1(1:end)))) (sum(L4(s1(1:end))))];
Qsm2 = [(sum(L1(s2(1:end)))) (sum(L2(s2(1:end)))) (sum(L3(s2(1:end)))) (sum(L4(s2(1:end))))];
Qsm3 = [(sum(L1(s3(1:end)))) (sum(L2(s3(1:end)))) (sum(L3(s3(1:end)))) (sum(L4(s3(1:end))))];
Qsm4 = [(sum(L1(s4(1:end)))) (sum(L2(s4(1:end)))) (sum(L3(s4(1:end)))) (sum(L4(s4(1:end))))];
Psm1 = Nm.*Qsm1;
Psm2 = Nm.*Qsm2;
Psm3 = Nm.*Qsm3;
Psm4 = Nm.*Qsm4;
Psm = [Psm1 Psm2 Psm3 Psm4];
 Pm1 = (Nm.*W)/maxStation;
 SSAL1 = sum(abs(Pm1 - Psm1)); % SSAL in Station 1
SSAL2 = sum(abs(Pm1 - Psm2));
SSAL3 = sum(abs(Pm1 - Psm3));
SSAL4 = sum(abs(Pm1 - Psm4));
 % Overall SSAL
SSAL = [SSAL1 SSAL2 SSAL3 SSAL4]/100
```

Solutions obtained for first stage balancing

```
maxStation = 4
nom = 4
not = 10
tw =    19    14    45     5    11    23    11    30    16    40
max_tw = 45
cmin = 53
ctMin = 53
sationInfo =
     1     2     2     1     3     1     3     3     4     4 and     1     2     2     1     3     1     4     3     3     4
Updated_Final_CYCLE_TIME = 59
```

MMALBP second stage balancing

```
s1 =      1     4     6
s2 =      2     3
s3 =      5     7     8
s4 =      9    10
time_s1 = 47
time_s2 = 59
time_s3 = 52
time_s4 = 56
Nm = 20    30    40    10
T1 = 4700
T2 = 5600
T3 = 5200
T4 = 5600
ST_TIME = 1.3056    1.5586    1.4444    1.5386
SSAL = 7.8500    4.1500    1.6500    5.3500
```

7. References

Askin, R.G. & Standridge, C.R. (1993). Modelling and analysis of manufacturing systems; John Wiley and Sons Inc, ISBN 0-471-51418-7

Baybars, L. (1986). A survey of exact algorithms for the simple assembly line balancing problems. *Management science*, Vol. 32, No. 8, (August, 1986), pp. (909-932)

Becker, C. & Scholl, A. (2006). A survey on problems and methods in generalized assemblyline balancing, *European journal of operational research,* Vol. 168, Issue. 3 (February, 2006), pp. (694–715), ISSN 0377-2217

Boysen, N., Fliedner, M. & Scholl, A. (2006). A classification of assembly line balancing problems. *European journal of operational research*, Elsevier, Vol 183, No. 2 (December, 2007), pp. (674–693)

Gu, L., Hennequin, S., Sava, A., & Xie, X. (2007). Assembly line balancing problem solved by estimation of distribution, Proceedings of the 3rd Annual IEEE conference on automation science and engineering scottsdale, AZ, USA

Papadopoulos, H.T; Heavey, C. & Browne, J. (1993). Queuing Theory in Manufacturing Systems Analysis and Design; Chapman & Hall, ISBN 0412387204, London, UK

Ponnambalam, S.G., Aravindan, P. & Naidu, G.M. (1999). A comparative evaluation of assembly line balancing heuristics. *International journal of advanced manufacturing technology,* Vol. 15, No. 8 (July 1999), pp. (577-586), ISSN: 0268-3768

Rekiek, B. & Delchambre, A. (2006). Assembly line design, the balancing of mixed-model hybrid assembly lines using genetic algorithm; Springer series in advance manufacturing, ISBN-10: 1846281121, Cardiff, UK

Rubinovitz, J. & Levitin, G. (1995). Genetic algorithm for assembly line balancing, *International Journal of Production Economics, Elsevier*, Vol. 41, No. 1-3 (October, 1995), pp (343-354), ISSN 0925-5273

Sabuncuoglu, I., Erel, E. & Tanyer, M. (1998). Assembly line balancing using genetic algorithms. *Journal of intelligent manufacturing*, Vol. 11, No. 3 (June, 2000), pp. (295-310), ISSN: 0956-5515

Scholl, A. (1993). Data of assembly line balancing problems. Retrieved from http://www.wiwi.uni-jena.de/Entscheidung/alb/, last accessed: 07 February 2008

Scholl, A. (1999). Balancing and sequencing of assembly lines, Second edition, Heidelberg: Physica, 318S, DM98,00, ISBN: 3790811807

Suresh, G. & Sahu, S. (1994). Stochastic assembly line balancing using simulated annealing, *International journal of production research*, Vol. 32, No. 8, pp. (1801-1810), ISSN: 1366-588X (electronic) 0020-7543 (paper)

Tasan, S.O. & Tunali, S. (2006). A review of current application of genetic algorithms in assembly line balancing, *Journal of intelligent manufacturing*, Vol. 19, No. 1 (February, 2008), pp. (49-69), ISSN: 0956-5515

Vilarinho, P.M. & Simaria, A.S. (2006). ANTBAL: An ant colony optimization algorithm for balancing mixed-model assembly lines with parallel workstations, *International journal of production research,* Vol 44, Issue 2, pp. 291–303, ISSN ISSN: 1366-588 0020-7543