Efficient Application of Metaheuristic Algorithms for Balancing Human Robot Collaborative Assembly Lines

A REVIEW 1 PROJECT REPORT

Submitted in partial fulfillment for the award of the degree of

B.Tech

in

Mechanical Engineering

by

Ranganathan S V – 18BME0374

Sridharan A P – 18BME0510

School of Mechanical Engineering

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

FEBRUARY & 2022

TABLE OF CONTENTS

ABSTRACT

The project aims to balance and improve the efficiency of different assembly line tasks that involve collaborative robots working hand in hand with humans. This is to be done through development of optimization algorithms by varying current metaheuristic algorithms, with an intention of less complex data processing and increased efficiency. Parallely, during the fine tuning of the developed algorithm, multi-objectives (cycle-time reduction, ergonomic & safety enhancement, energy consumption reduction) are considered along with real life assembly line constraints. Moreover, the developed algorithms are to be consecutively compared with several benchmark functions and datasets and their relative performance evaluated.

CHAPTER 1

INTRODUCTION

Assembly line in the manufacturing process comes at last which delivers the finished product with a sequence of several tasks. Normally these tasks were performed by humans earlier before the introduction of robots. However, due to ever-changing consumer tastes and to outperform the competitors, many industries started using robots in their assembly line. The main reasons for this change are to improve production rate, efficiency, reduce cost(wastes) and to increase agility towards changing demands.

This use of robots in assembly lines was called as **robotic assembly lines(RAL)**. Use of robots in industries helped them to reduce labour cost, variability in production, and enhance productivity as robots can work 24/7 all year with no fatigue or illness.

Advancements in technology helped this evolution to further replace humans by robots in performing complex tasks too. Since the introduction of robots in assembly lines, **robotic assembly line balancing(RALB)**, and extension of ALB have become of interest for many researchers and industries to further optimize their production. This varied from the conventional ALBP as here two different problems are to be addressed and they are, assignment of tasks to the given workstations and assignment of different available robots to the workstations.

The general assembly line balancing problem has some objective such as minimising cycle time, cost, etc.., with some constraints such as precedence relation of tasks, number of workstations, etc.., As the robots are accurate in performing tasks, their task times are assumed to be deterministic while solving RAL BP. Some other assumptions made by the researchers on solving them were,

- The precedence relations of tasks are known and the task times depend completely on the type of robot chosen.
- One workstation comprises only one robot and all types of robot are always ready to use without any capacity and cost limitations.

Further RALBP is classified into several types such as,

- ➔ RALBP TYPE I – Aims at reducing the number of workstations by assigning best suited robots and tasks to workstations.
- ➔ RALBP TYPE II – Aims at minimising cycle time with a predefined number of workstations.
- ➔ RALBP TYPE E – Aims at reducing both cycle time and workstations, thus maximising the assembly line efficiency.
- ➔ RALBP TYPE F – Aims at finding feasible solutions for a predefined set of workstations and cycle times.
- ➔ RAL BP TYPE COST – Aims at the monetary and economic aspects of RAL.
- ➔ RAL BP TYPE O – Other categories are classified in this type.

In RALB, the classifications are based on the structure of the assembly lines**[AL - assembly line]**. They are MAL(multi manned), PWAL(parallel workstation), PAL(parallel), UAL(U-shaped), StAL(straight line), 2SAL(two sided), PUL(parallel U-line), PAUL(parallel adjacent U-line), PMAL(parallel multi-manned).

As the automation in the manufacturing department is increasing significantly, several robots are developed to carry out several tasks to increase the production rate. However, not all tasks can be automated due to the lack of flexibility with available robotic technologies. Humans on the other hand are flexible, adaptable according to market demands, and also have decision making skills with creativity. To remain relevant, many manufacturing enterprises show interest in human robot collaboration where the skills of humans and the speed, endurance, and accuracy of robots are brought together to increase production. This also helps the human workers as the repetitive and stressful tasks are done by robots.

Many algorithms and meta-heuristics were used to find the best allocation of robots and humans to workstations, determine the optimal task sequence, cycle time, etc.., Some are,

1) Artificial Bee Colony Algorithm (ABC)
2) Genetic Algorithm (GA)
3) Differential Evolution (DE)
4) Teaching Learning based optimization (TLBO)
5) Particle swarm optimization (PSO), etc..,

CHAPTER 2

LITERATURE REVIEW

[1] Paper contains all types of RALB problem approaches to date including classifications, solution methods, layout considerations.

[2] Robot ergonomic considerations like task suitability and precision factors, working environment and runtime, orientation in workstations were referred to as our objective problems.

[3] RALB problem solution approaches and considerations that weren't accounted for previously.

[4] Puts forth every type of assembly line problem solution approach and work considerations. All algorithms used till date and their suitabilities are given.

[5] Puts forth three different layouts for the human-robot assembly line (only human/both in a workstation/both in different workstations). An MILP approach to solve every scenario was formulated, which was referred to

as our problem synthesis.

[6] Tabulates all possible risks in a human-robot collaborative assembly line environment, along with precedence insights of every risk. Risks are referred to effectively formulate our problem with the objective of increasing ergonomics.

[7] Introduction of new gaussian mutation throw point strategy to memorize all the non-dominated solutions (pareto optimal solutions in the population); Various other related strategies.

[8] Insights on commands for every arithmetic operation involved in PSO equations. Was used to gain an understanding to deal with problems that need to maintain the population members' values within context.

[9] Variant models of IPSO with three non-linear time-varying strategy introductions; MPSO with asymmetric time varying trend for acceleration coefficient adjustment. Both for biasing the pace of the coefficient 'c'.

[10] Review of variants like Comprehensive learning PSO (CPSO), Extraordinary PSO (EPSO), Heterogeneous comprehensive learning PSO (HCPSO), Darwinian PSO (DPSO), fractional-order DPSO (FDPSO), improved random drift PSO (IRDPSO) algorithms.

[11] Variant mTLBO, introduced the concept of tutorial class in the Learner Phase and its implementation. DLTLBO, where a neighborhood search of the teacher phase is added to produce a new mutation vector, thereby taking advantage of differential learning (hybrid).

[12] Balanced TLBO (BTLBO), introduced two new phases; tutoring and restarting, providing a balance between exploitation and exploration capabilities. Emphasis on increasing the possibility of finding the global optima.

[13] Task allocation algorithm using TLBO for a real time two-sided assembly line model. Referred to the mixed model approach used while line balancing.

[14] Referred as an additional material for validating the fact of non TLBO usage in robotic assembly lines.

[15] Basic approach in using TLBO to solve a two-sided assembly line balancing problem. Referred to the constraints considered for two/one side distinction.

CHAPTER 3

GAPS IN THE LITERATURE

- Since the study started on RAL BP, only 33 articles have been published to date since 1991(over 30 years). The research on this field was less and skyrocketed recently (2015-2019).
- In 1997, most of the researchers started using deterministic variable task times that were determined by the types of robots. This was due to the development in technology that resulted in the invention of many different robots that can perform all the tasks.
- Mathematical models were used to describe the relations between the different objectives and to handle the constraints effectively. As the size of the problem increased, meta-heuristics started dominating the other methods. Many meta-heuristics were developed to tackle this NP-hardness of the RALBP.
- Introduction of Cobots(collaborative robots) to the assembly line created a new trend in the research area of RALBP. This use of cobots in the assembly line showed better results and gained interest by many industries. **As this area of research is quite new, only a few papers have been published since 2019.**
- Many real life constraints such as tools assignment, tool space optimization, etc.., were not addressed in most of the papers and this could be the future in the study of RAL BP.

- Almost all the research on RALBP assumes a fully automated assembly line which requires huge investments. Not all the industries are capable of making such investments and still many of the industries slowly started to introduce robots in their assembly lines. So the **study of human robot collaboration assembly lines could be a great research area as many industries opt for them as they are easy and require less investment.**

- The studies that included cobots in the assembly line mostly preferred only one robot in a workstation. Research study on multiple robots and humans working in parallel could be a great area of research. As humans started working alongside robots, study of ergonomics and real life constraints such as human fatigue, sequence dependent setup times, tool and accessory changing, part transportation between stations, etc.., can be included in RABLP as no one has ever done that previously due to its high complexity.

- Most industries focus on only one objective while designing assembly lines rather than many.

Research on multi-objective optimizations could be more relevant to real life problems. As the introduction of multi-objectives created more complexity, many meta-heuristics were developed. From the recent studies, hybrid metaheuristics showed better results than meta heuristics. So study on hybrid metaheuristics on solving multiple objectives could create a way for finding better pareto optimal solution sets.

CHAPTER 4

PROBLEM DEFINITION

This problem type is classified as a multi **objective RAL BP TYPE - II** that focuses on minimising cycle time and energy consumption with a given number of workstations. The two algorithms that we focus on modifying (make some changes to fit our problem statement) are,

- ❖ Particle swarm optimisation **(PSO)**
- ❖ Teaching learning based optimisation **(TLBO)**

WHY PSO and TLBO a GREAT CHOICE TO OUR PROBLEM:

The problem that we considered is solving a multiobjective human robot collaborative assembly line balancing.

TLBO being a new algorithm, is yet to be explored by many researchers and this gives novelty to our paper.

TLBO and PSO can be modified easily according to the problem as it has very few parameters yet more combinations that influence the output.

Many new variants can be created from TLBO and PSO as the concept is simple and easy to implement.

TLBO and PSO can be merged with other popular meta-heuristics such as Differential evolution, Genetic algorithm, etc.., and a hybrid algorithm can be created, which uses the simplicity of TLBO and PSO with the marginal advantages of other popular algorithms.

CHAPTER 5

OBJECTIVES

- To develop metaheuristic algorithms for balancing and scheduling of Human-robot collaborative assembly line operations.
- Considering the assembly line structure of StAL (straight line assembly line) where a single product is assembled with a known number of workstations, to use the developed metaheuristic algorithms to optimize objectives of CYCLE TIME REDUCTION, IMPROVEMENT OF ERGONOMICS, REDUCING OVERALL LINE ENERGY CONSUMPTION.
- To consider setup times, sequence dependent times and other real-time line constraints that haven't been considered to solve human-robot ALB problems so far.

CHAPTER 6

WORK CARRIED OUT SO FAR

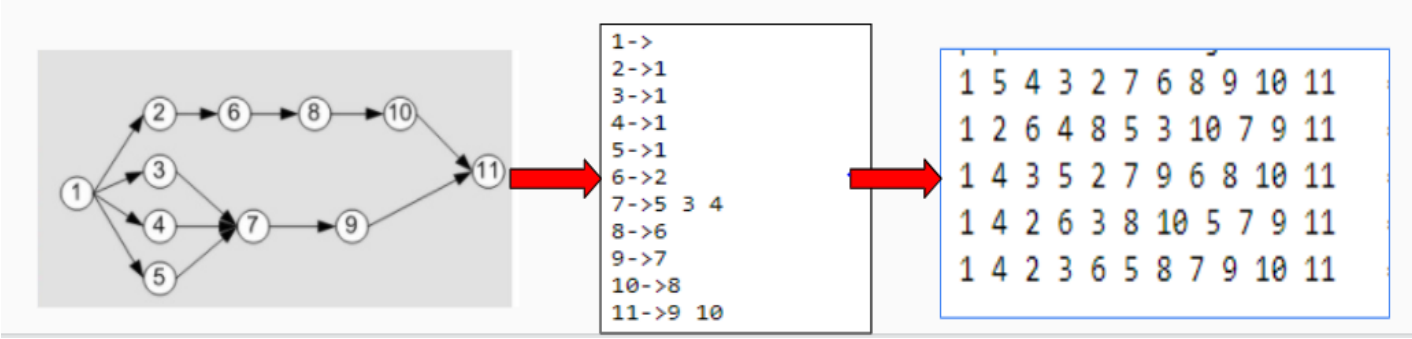
Literature Survey - We conducted an extensive literature survey and studied various assembly line balancing techniques of only workforce, robotic lines as well as human-robot lines. Parallely, metaheuristic algorithms and their benchmark function efficiencies were reviewed to decide the best algorithm to proceed. We have finalized to code our own variant algorithm of PSO and TLBO, suited to our problem environment and considerations.

6.1 SEQUENCE GENERATION:

For generating sequences, a precedence matrix is generally used in many papers.
But in our proposed method, we use an adjacency list that stores the precedence relation efficiently and helps in generation of sequences that satisfy these relations.
The algorithm is as:

- First an available set is created with the tasks that have no precedence tasks.
- Then one task is selected randomly from that set and the tasks following it are added in the available set.
- This step is repeated until all the tasks are assigned and the available set is empty.

The use of adjacency lists is very helpful in implementing this algorithm.
Adjacency holds the data as shown in this example.



6.2 OBJECTIVE FUNCTION EVALUATION (cycle time):

For calculating the cycle time of a particular sequence generated, a consecutive method algorithm is used.
The working of this algorithm is:

- Cycle time is initially fixed with a value by summing up the minimum task time of all robots.
- For each station, the sequence is tried with all the available robots with a given number of stations and the cycle time that was found previously.
- In each iteration, if the found cycle time is not feasible, the cycle time is incremented.
- This step is repeated until a feasible cycle time is found which is the optimal cycle time for the given sequence.

For example, the image on the right shows an example of allocation of tasks in each workstation using the consecutive method.
As we can see here, the overall cycle time of such allocation is maximum cycle time, i.e.., **202** for the sequence **[1, 5, 3, 4, 7, 9, 2, 6, 8, 10, 11]** with **4 workstations.**

```
station number 1's allocation:
robot assigned : 3
tasks assigned : 1 5 3
its cycle time : 201

station number 2's allocation:
robot assigned : 4
tasks assigned : 4 7 9
its cycle time : 202

station number 3's allocation:
robot assigned : 4
tasks assigned : 2 6
its cycle time : 152

station number 4's allocation:
robot assigned : 2
tasks assigned : 8 10 11
its cycle time : 202
```

6.3 FORMULATION OF OUR VARIANT ALGORITHMS OF PSO & TLBO:

For implementing the standard PSO & TLBO for a discrete problem, some assumptions and formulations are made such as,

- Population matrix consists of feasible sequences.
- Velocity matrix consists of randomly generated numbers ranging from [0 - no_of_tasks].
- A new counter array is used that stores an integer that indicates the number of times the solution is retained in the population.
- A new best_solution (variable) is used to keep track of the best solution removed from the population in the last phase of the algorithm.
- Updation of the swarm is done according to the rules that are mentioned in upcoming slides.
- For encoding purposes, only the sequence array is used.
- For decoding purposes, the robots assigned and the tasks performed in each station are determined by knowing the optimal cycle time.
- The addition of setup times and sequence dependent time is used to make the problem more suitable for industrial purposes.
- Setup time is added with the task time for each task performed by the robot.
- Sequence-dependent time is added only for the tasks that are not first in the station as these times are dependent on the previously performed tasks(the first performed task in each station by the robot has no previous task).

CHAPTER 7

METHODOLOGY

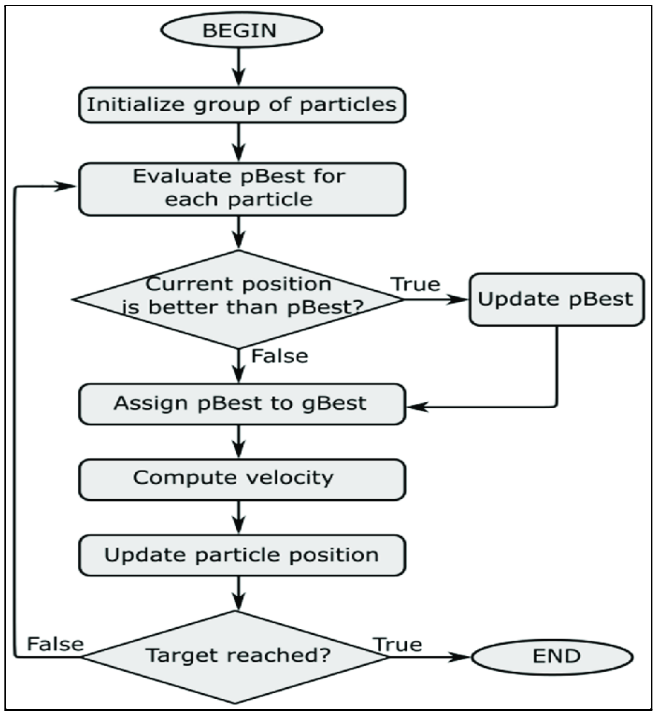
7.1 WORKING OF MODIFIED DPSO:

$$X_i^{t+1} = X_i^t + V_i^{t+1}$$

$$V_i^{t+1} = c_1 V_i^t + c_2(Pbest_i^t - X_i^t) + c_3(Gbest_t - X_i^t)$$

NOTATIONS USED:

t = indicates the iteration number
Xi = a single i'th solution in the population
Vi = a single i'th velocity vector in the population
P_best_i = indicates the best solution obtained by i'th particle so far
G_best = indicates the best solution obtained by the whole swarm so far
c1 = inertia weight
c2 = acceleration coefficient of P_best
c3 = acceleration coefficient of G_best



POPULATION UPDATION:

For implementing the modified DPSO method, the velocity vector is taken as an array of integers ranging from [0 - no.of.tasks] instead of velocity pairs that are used in most of the papers.
These velocity vectors are used to update the population using certain conditions. They are,

SUBTRACTION (position - position)

Let X1,t = [x1,1, x1,2, x1,3, x1,4, x1,5, x1,6, x1,7]
X2,t = [x2,1, x2,2, x2,3, x2,4, x2,5, x2,6, x2,7]
V1,t =X1,t - X2,t
In this case, if x1 and x2 in the jth position are equal, then v1= 0. Otherwise, v1=x1. For example,

```
X1,t = 2 3 1 4 5
X2,t = 1 3 5 4 2
V1,t = X1,t - X2,t = 2 0 1 0 5
```

ADDITION (position + velocity)

If the jth element of velocity (vj) is equal to 0, the jth position value (xj,t) is inserted into the jth element of the new position (xj,t+1).
In the meantime, if vj is nonzero and does not appear in the new position, then xj,t+1=vj. Otherwise, xj,t+1 is equal to 0. For example,

```
X1,t = 3 2 5 1 4
V1,t = 0 2 0 3 4
X1,t+1 = 3 2 5 0 4
```

ADDITION (velocity + velocity)

For new velocity, V=V1 + V2, the jth element of V can be derived as follows,
Vj = v1,j IF v1,j!=0 && v2,j!=0
Vj = v1,j IF v1,j!=0 && v2,j==0
Vj = v2,j OTHERWISE
For example,

```
V1,t = 3 2 0 1 0
V2,t = 0 1 5 3 0
Vt+1 = 3 2 5 1 0
```

MULTIPLICATION (coefficient * velocity)

This operation can be represented as V2= c * V1, where coefficient c lies in[0, 1] is used to control the effect of V1 that inherits in V2.
For this purpose, a random number, r that lies in [0, 1] is generated. IF r< c, v2=v1, ELSE v2= 0.
For example,

```
V1 = 2 4 1 3 5, C = 0.4
R = .1 .7 .3 .9 .3
V2 = 2 0 1 0 5
```

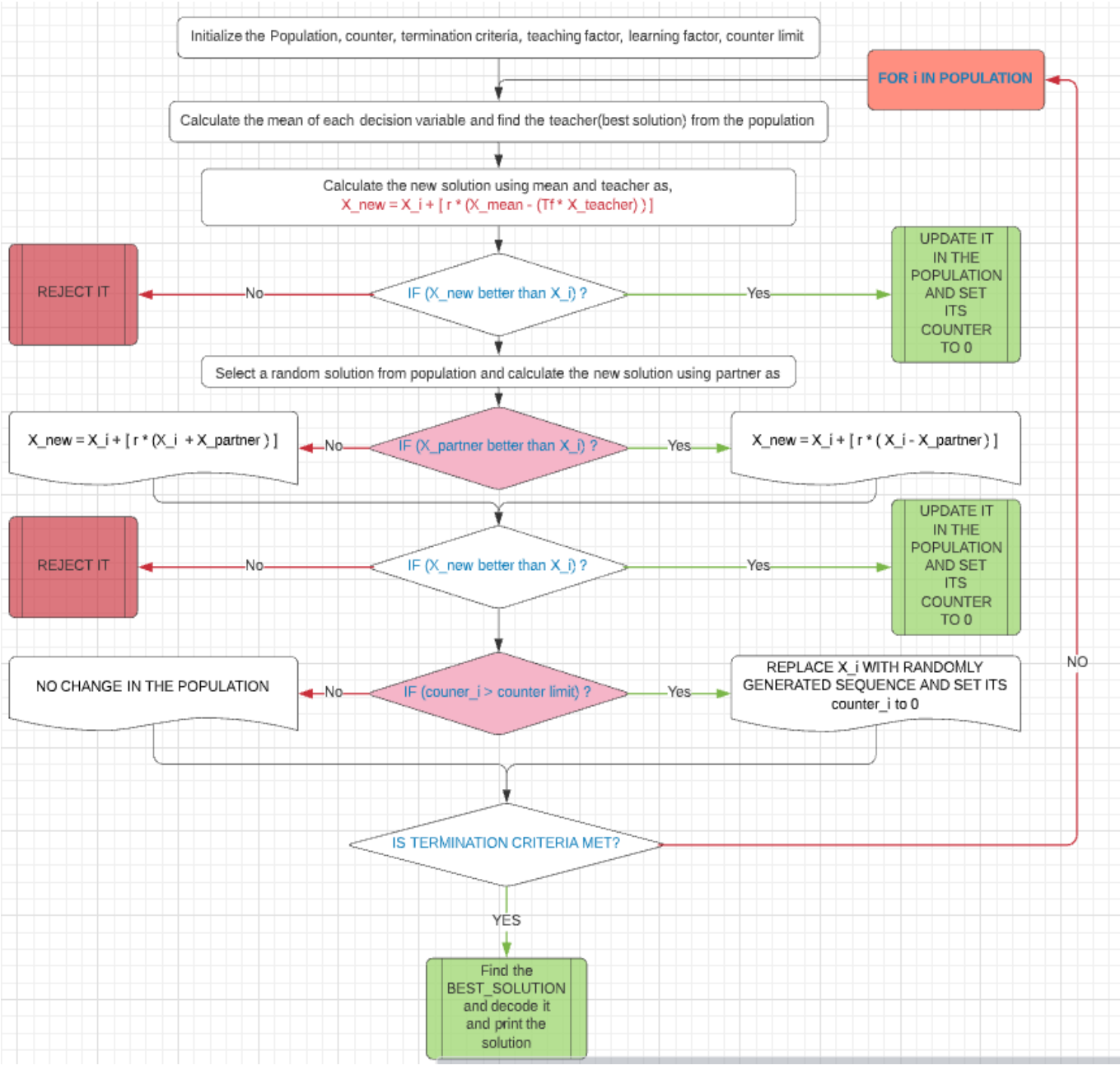
REPAIR FUNCTION

This function is used to repair a sequence which does not follow the precedence relation.
This does multiple swaps and replacements so that the given sequence is in bounds with the precedence relation.
For example,

```
X = 0 3 1 4 5
new_X = 1 2 3 4 5
```

7.2 WORKING OF HYBRID TLBO:

PROPOSED FLOWCHART:



NOTATIONS USED:

X_i = a single i'th solution in the population.
r = Coefficient that indicates the impact of the mean of the population and the teacher
Tf = Teaching coefficient that indicates the impact of teacher
X_new = newly generated solution for the ith solution in the population
X_mean = mean of the population
X_best = teacher (best solution in the population)
X_partner = mean of the population

CALCULATING THE MEAN OF THE POPULATION

- As mentioned in the previous slide, in teacher phase calculation of the mean of the population is used in the equation, **X_new = X_i + r * [X_mean - (Tf * X_best)].**
- As the traditional way of calculating the mean can't be used here (discrete optimization problem), a different methodology is used here.
- The calculation is done as,
  - ➔ First the average of the task numbers in that j'th location in the population is calculated.
  - ➔ If the calculated average is not already present in the mean sequence, then it is pushed into the mean sequence.
  - ➔ If the calculated mean value is already present in the mean sequence, then the closest value to the mean in that j'th location of the population is inserted if it is not present in the mean sequence.
  - ➔ Otherwise, a random number is generated within the range of 1 - n (number of tasks) and is inserted in the mean sequence if it is not already present in the mean sequence.

SUBTRACTION OPERATOR

- As mentioned previously, in teacher phase and learner phase, subtraction of the solutions is done as,
  X_new = X_i + r * [X_mean - (Tf * X_best)].
- As the traditional way of subtraction is not possible here as the variables are discrete, a different methodology is used as,

  - ➔ For each task number in the i'th position of the sequences, if both the sequence have the same task number, then 0 is inserted in the i'th position of the result.
  - ➔ If they are not the same, a random number is generated between 0 and 1 and is compared with the coefficient(Tf / Tp).
  - ➔ If the random number generated is less than the coefficient then, the task number in the i'th position from the second sequence is inserted into the i'th position of the result.
  - ➔ Otherwise, 0 is inserted in the i'th position of the result.

ADDITION OPERATOR

- As mentioned previously, in the teacher phase and learner phase, addition of the solutions is done as,

  X_new = X_i + r * [X_mean - (Tf * X_best)].

- As the traditional way of addition is not possible here as the variables are discrete, a different methodology is used as,

  - For each task number in the i'th position of the sequences, if both the sequence have the same task number, then the task number from the i'th position of the first sequence is inserted in the i'th position of the result.
  - If they are not the same, a random number is generated between 0 and 1 and is compared with the coefficient(r).
  - If the random number generated is less than the coefficient, then the task number in i'th from the second sequence is inserted into the i'th position of the result.
  - Otherwise, 0 is inserted in the i'th position of the result.

CHAPTER 8

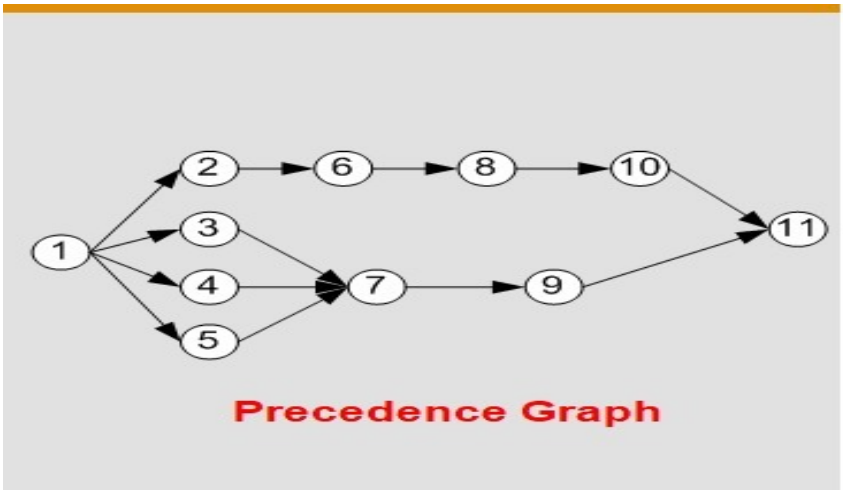
RESULTS & DISCUSSIONS

8.1 DATA USED:

TASK TIME AND PRECEDENCE GRAPH:

Activities	Performance Time			
	Robot 1	Robot 2	Robot 3	Robot 4
1	81	37	51	49
2	109	101	90	42
3	65	80	38	52
4	51	41	91	40
5	92	36	33	25
6	77	65	83	71
7	51	51	40	49
8	50	42	34	44
9	43	76	41	33
10	45	46	41	77
11	76	38	83	87

Performance times of 11 Activities by 4 Robots



SETUP TIMES:

ACTIVITIES

ROBOT

18	15	13	18	16	16	10	11	17	14	17
10	11	16	16	14	16	10	18	14	13	20
19	14	13	18	19	18	20	14	19	17	17
11	13	13	15	11	12	11	13	19	12	14

This is the input data used that represents the setup times of 4 robots for given 11 different activities.
This setup time is added with the task time for all the robots that perform that particular task. This setup time includes the time for setting up the robot, fixing the tool, etc..,

SEQUENCE DEPENDENT TIMES CONSIDERED FOR ALGORITHM SYNTHESIS:

ROBOT 1

```
{0, 19, 20, 16, 11, 20, 12, 18, 12, 19, 18},
{10, 0, 10, 14, 11, 10, 17, 15, 16, 10, 10},
{10, 10, 0, 12, 17, 19, 20, 15, 12, 15, 12},
{11, 19, 11, 0, 11, 19, 20, 18, 17, 20, 14},
{17, 13, 15, 17, 0, 10, 10, 18, 19, 10, 16},
{15, 12, 11, 11, 20, 0, 14, 12, 16, 13, 11},
{18, 10, 10, 14, 20, 18, 0, 19, 11, 15, 10},
{15, 20, 20, 13, 10, 16, 11, 0, 11, 14, 11},
{10, 15, 19, 15, 17, 20, 19, 19, 0, 16, 19},
{19, 10, 17, 16, 19, 15, 16, 14, 15, 0, 11},
{13, 13, 20, 17, 12, 19, 18, 16, 18, 17, 0}
```

ROBOT 2

```
{0, 11, 16, 13, 20, 12, 11, 14, 10, 20, 12},
{17, 0, 19, 10, 11, 10, 13, 20, 12, 14, 12},
{11, 10, 0, 20, 17, 18, 15, 13, 14, 11, 12},
{18, 14, 11, 0, 14, 13, 10, 13, 13, 15, 19},
{12, 13, 18, 12, 0, 17, 14, 10, 17, 14, 11},
{19, 13, 18, 14, 19, 0, 17, 18, 10, 14, 20},
{20, 14, 13, 13, 12, 17, 0, 15, 13, 16, 19},
{20, 17, 13, 14, 10, 11, 20, 0, 12, 16, 16},
{18, 19, 12, 14, 15, 20, 14, 19, 0, 12, 10},
{19, 13, 10, 13, 17, 16, 16, 13, 12, 0, 20},
{13, 16, 18, 14, 15, 10, 17, 10, 16, 12, 0}
```

ROBOT 3

```
{0, 15, 11, 15, 19, 10, 20, 10, 19, 18, 11},
{20, 0, 19, 13, 15, 20, 13, 18, 17, 16, 13},
{14, 20, 0, 12, 14, 15, 16, 15, 13, 11, 20},
{15, 16, 16, 0, 14, 15, 20, 11, 16, 19, 12},
{14, 20, 17, 13, 0, 12, 18, 16, 14, 12, 14},
{18, 12, 18, 12, 19, 0, 14, 20, 20, 19, 14},
{16, 20, 18, 19, 19, 19, 0, 16, 19, 19, 15},
{13, 20, 14, 14, 18, 20, 18, 0, 13, 15, 18},
{20, 18, 16, 20, 11, 16, 18, 18, 0, 11, 15},
{15, 20, 13, 11, 13, 17, 19, 20, 10, 0, 18},
{14, 15, 13, 13, 10, 19, 15, 14, 16, 12, 0}
```

ROBOT 4

```
{0, 20, 10, 16, 17, 16, 11, 18, 10, 14, 15},
{11, 0, 17, 19, 11, 16, 17, 13, 11, 10, 16},
{14, 11, 0, 10, 14, 19, 17, 16, 18, 17, 16},
{18, 10, 10, 0, 11, 17, 20, 15, 11, 19, 18},
{16, 15, 20, 10, 0, 10, 11, 11, 14, 14, 11},
{19, 14, 15, 17, 19, 0, 14, 15, 13, 20, 15},
{13, 10, 16, 18, 18, 19, 0, 17, 17, 15, 11},
{16, 15, 13, 14, 15, 12, 18, 0, 11, 17, 10},
{17, 13, 18, 14, 16, 12, 17, 15, 0, 20, 14},
{20, 18, 11, 19, 15, 18, 13, 18, 18, 0, 11},
{19, 10, 14, 19, 18, 10, 10, 12, 19, 15, 0}
```

8.2 RESULTS OF MODIFIED DPSO vs HYBRID TLBO:

RESULTS OF MODIFIED DPSO:

```
ITERATION 1 :

population = objective function value(cycle_time) :
1 3 2 4 6 8 10 5 7 9 11   =  223
1 5 2 4 6 3 7 9 8 10 11   =  207
1 2 4 6 3 8 10 5 7 9 11   =  229
1 4 3 5 7 2 9 6 8 10 11   =  208
1 3 4 2 5 6 8 10 7 9 11   =  208

velocity vectors of each particle :
7 1 2 3 11 8 10 5 9 6 4
6 8 3 5 1 11 10 9 7 4 2
3 9 5 1 7 6 8 2 10 4 11
4 2 8 5 7 10 6 1 11 3 9
4 11 5 7 3 9 1 10 2 6 8

p_best values for each particle :
1 3 2 4 6 8 10 5 7 9 11 = 223
1 5 2 4 6 3 7 9 8 10 11 = 207
1 2 4 6 3 8 10 5 7 9 11 = 229
1 4 3 5 7 2 9 6 8 10 11 = 208
1 3 4 2 5 6 8 10 7 9 11 = 208

g_best for the whole swarm :
1 5 2 4 6 3 7 9 8 10 11 = 207
```

```
ITERATION 10 :

population = objective function value(cycle_time) :
1 5 4 2 6 8 10 3 7 9 11   =  221
1 5 3 4 7 9 2 6 8 10 11   =  202
1 4 5 3 2 7 9 6 8 10 11   =  207
1 2 6 8 5 10 3 4 7 9 11   =  223
1 3 2 4 5 6 7 8 9 10 11   =  211

velocity vectors of each particle :
0 1 2 3 11 8 10 5 9 6 4
0 8 3 5 1 11 10 9 7 4 2
0 9 5 1 7 6 8 2 10 4 11
0 2 8 5 7 10 6 1 11 3 9
0 11 5 7 3 9 1 10 2 6 8

p_best values for each particle :
1 5 4 2 6 8 10 3 7 9 11 = 221
1 5 3 4 7 9 2 6 8 10 11 = 202
1 4 5 3 2 7 9 6 8 10 11 = 207
1 4 3 5 7 2 9 6 8 10 11 = 208
1 4 3 2 5 6 7 8 9 10 11 = 208

g_best for the whole swarm :
1 5 3 4 7 9 2 6 8 10 11 = 202
```

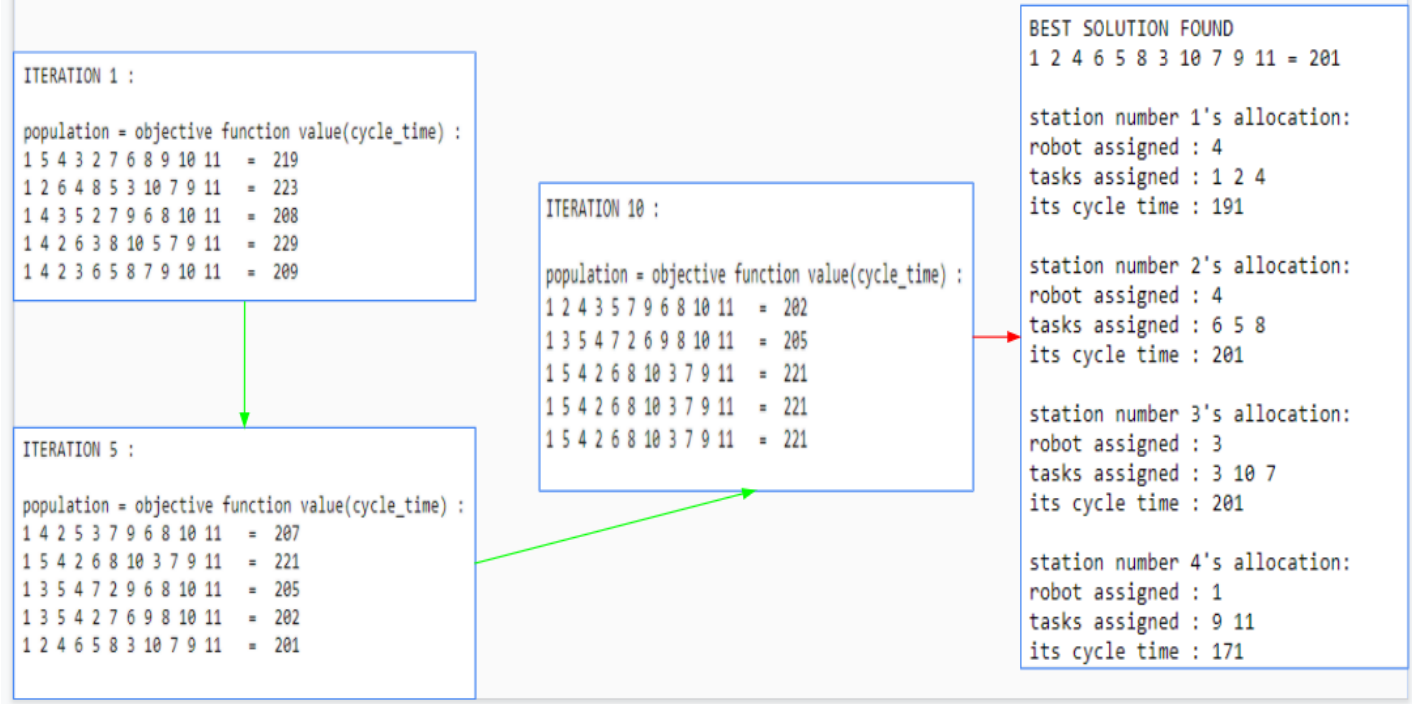
```
station number 1's allocation:
robot assigned : 3
tasks assigned : 1 5 3
its cycle time : 201

station number 2's allocation:
robot assigned : 4
tasks assigned : 4 7 9
its cycle time : 202

station number 3's allocation:
robot assigned : 4
tasks assigned : 2 6
its cycle time : 152

station number 4's allocation:
robot assigned : 2
tasks assigned : 8 10 11
its cycle time : 202
```

RESULTS OF HYBRID TLBO:



COMPARISON OF MODIFIED DPSO & HYBRID TLBO:

Initialize a random population, velocity vector, p_best and g_best

For t=1 to T (termination criteria)

For i=1 to P (population size)

%UPDATE VELOCITY

Vi_t+1 = c1 * Vi_t + c2 * (p_best – Xi_t) + c3 * (g_best – Xi_t)

%UPDATE POSITION

Xi_t+1 = Xi_t + Vi_t

IF Xi_t+1 better than p_best

Update p_best

IF Xi_t+1 better than g_best

Update g_best

END

END

Initialize a random population and BEST_SOLUTION (holds the answer)

For t=1 to T (termination criteria)

For i=1 to P (population size)

%TEACHING PHASE%

Find the best solution from entire population (X_best)

Determine mean of each decision variable (X_mean)

X_new = X_i + r * [X_mean – (Tf * X_best)]

IF X_new better than X_i update it, ELSE reject it

%LEARNING PHASE%

Choose any solution from population randomly (X_partner)

IF X_partner better than X_i

X_new = X_i + r * (X_i – X_partner)

ELSE

X_new = X_i + r * (X_i + X_partner)

IF X_new better than X_i update it, ELSE reject it

%PEER PRESSURE OVERCOMING PHASE%

IF counter_i > counter_limit

IF X_i better than BEST_SOLUTION update it

Update X_i with randomly generated sequence

Counter_i = 0

END

END

M-DPSO	H-TLBO
1.Modification of traditional Particle Swarm Optimisation to fit into solving discrete optimisation(DO) problems. **2.**Number of times objective function evaluated is, **Np*T** (Np = population size). **3.**NO GREEDY selection to update population. **4.**Number of parameters that influence the output is, **c1, c2, c3**. **5.**Number of variables in the population updation equation, **Velocity, G_best, P_best**. **6.**Number of special operators used in equations, Addition operator, Subtraction operator, Multiplication operator, Repair function. **7.**Best solution obtained is present in the population matrix. **8.**Best solution obtained for the example problem, Optimal cycle time = **202**.	**1.**Modification of traditional TLBO which combines the advantages of scout phase from ABC algorithm and simplicity of TLBO to solve DO problems. **2.**Number of times objective function evaluated is, **Np*T + Np*T** (T = number of iterations). **3.**Uses GREEDY selection to update population. **4.**Number of parameters that influence the output is, **r, Tf/Tp**. **5.**Number of variables in the population updation equation, **X_mean, X_best, X_partner**. **6.**Number of special operators used in equations, Addition operator, Subtraction operator, Mean calculation, Repair function. **7.**Best solution obtained may or may not be present in the population matrix. **8.**Best solution obtained for the example problem, Optimal cycle time = **201**.

CHAPTER 9

WORK TO BE DONE

For the next review:

1. To implement Multi-objective in reducing energy consumption along with cycle-time reduction
2. To develop the code to suit the user's flexibility to fine tune the assembly line parameters as input and get the output data accordingly.
3. To include both human & robotic data sets and allocation, according to user/referred real world constraints.
4. Possibly arrive at more new variants in both PSO and TLBO, after inclusion of the above considerations.

CHAPTER 10

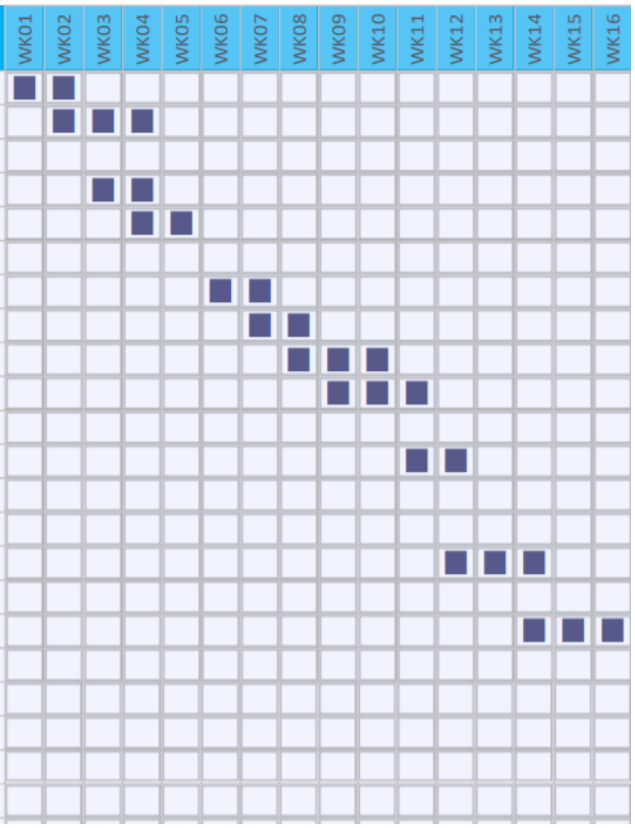
MILESTONES AND GANTT CHART (WORK PLAN)

REVIEW 1: Task ID 01-03.2
REVIEW 2: Task ID 04-08
REVIEW 3: Task ID 09-10

Project Description
"Efficient application of optimization algorithms for balancing human robot collaborative assembly lines" The projects aims to balance and improve the efficiency of different assembly line tasks that involve 'cobot' or collaborative robots working hand in hand with humans through optimization algorithms.

Remarks
1. Overlapping of tasks is to carry out to interlink related work parallely. 2. Timeline mentioned are the expected ones. Actual work is to be done sooner. 3. Any new tasks will be updated in the upcoming reviews.

Task ID	Task Name	Start Week	End Week	Resources
01	Define problem objectives	WK01	WK02	
02	Literature review & gap anly	WK02	WK04	
03	SINGLE OBJECTIVE models			
03.1	Coding PSO variants	WK03	WK04	
03.2	Coding TLBO variants	WK04	WK05	REVIEW 1
04	Multi-objective(MO) plannin	WK06	WK07	
05	Data collection for H-R ALB	WK07	WK08	
06	Literature review & gap idnt	WK08	WK10	
07	Compiling MO H-R ALB data	WK09	WK11	
	in PSO/TLBO variant codes.			
08	Fine tuning of compiled	WK11	WK12	
	variants.			REVIEW 2
09	Result comparison & anlysis	WK12	WK14	
	with benchmark functions			
10	Formal report	WK14	WK16	REVIEW 3



REFERENCES

[1] Chutima, Parames. (2022). A comprehensive review of robotic assembly line balancing problem. Journal of Intelligent Manufacturing. 33. 10.1007/s10845-020-01641-7.

[2] Christian Weckenborg, Thomas S. Spengler, Assembly Line Balancing with Collaborative Robots under consideration of Ergonomics: a cost-oriented approach, IFAC-PapersOnLine, Volume 52, Issue 13, 2019, Pages 1860-1865, ISSN 2405-8963, https://doi.org/10.1016/j.ifacol.2019.11.473.

[3] Weckenborg, C., Kieckhäfer, K., Müller, C. *et al.* Balancing of assembly lines with collaborative robots. *Bus Res* 13, 93–132 (2020). https://doi.org/10.1007/s40685-019-0101-y

[4] Nils Boysen, Philipp Schulze, Armin Scholl, Assembly line balancing: What happened in the last fifteen years?, European Journal of Operational Research, 2021, ISSN 0377-2217, https://doi.org/10.1016/j.ejor.2021.11.043.

[5] Tamás Koltai, Imre Dimény, Viola Gallina, Alexander Gaal, Chiara Sepe, An analysis of task assignment and cycle times when robots are added to human-operated assembly lines, using mathematical programming models, International Journal of Production Economics, Volume 242, 2021, 108292, ISSN 0925-5273, https://doi.org/10.1016/j.ijpe.2021.108292.

[6] Nicole Berx, Wilm Decré, Ido Morag, Peter Chemweno, Liliane Pintelon, Identification and classification of risk factors for human-robot collaboration from a system-wide perspective, Computers & Industrial Engineering, Volume 163, 2022, 107827, ISSN 0360-8352, https://doi.org/10.1016/j.cie.2021.107827.

[7] Ying Sun, Yuelin Gao. (2019). A Multi-Objective Particle Swarm Optimization Algorithm Based on Gaussian Mutation and an Improved Learning Strategy. doi:10.3390/math7020148

[8] Mohd Fadzil Faisae Ab Rashid, Windo Hutabarat, Ashutosh Tiwari. (2016). Multi-objective discrete particle swarm optimisation algorithm for integrated assembly sequence planning and assembly line balancing. DOI: 10.1177/0954405416673095

[9] Parsopoulos, Konstantinos & Vrahatis, Michael. (2008). Multi-objective particle swarm optimization approaches. 10.13140/2.1.5189.4721.

[10] Kashani, A.R., Chiong, R., Mirjalili, S. *et al.* Particle Swarm Optimization Variants for Solving Geotechnical Problems: Review and Comparative Analysis. Arch Computat Methods Eng 28, 1871–1927 (2021). https://doi.org/10.1007/s11831-020-09442-0

[11] Y. Ma, X. Zhang, J. Song et al., A modified teaching-learning-based optimization algorithm for solving optimization problem, Knowledge-Based Systems (2020), doi: https://doi.org/10.1016/j.knosys.2020.106599.

[12] Ahmad Taheri, Keyvan RahimiZadeh, Ravipudi Venkata Rao. An efficient Balanced Teaching-Learning-Based optimization algorithm with Individual restarting strategy for solving global optimization problems, Information Sciences, Volume 576, 2021, Pages 68-104, ISSN 0020-0255, https://doi.org/10.1016/j.ins.2021.06.064.

[13]Hamzadayi, A. (2018). Balancing of mixed-model two-sided assembly lines using teaching-learning based optimization algorithm. *Pamukkale University Journal of Engineering Sciences, 24*, 682-691.

[14]Li, Da & Zhang, Chaoyong & Shao, Xin & Xie, Zhan. (2014). An Improved TLBO Algorithm for Balancing Stochastic Two-Sided Assembly Line. Applied Mechanics and Materials. 610. 345-349. 10.4028/www.scientific.net/AMM.610.345.

[15]Qiuhua Tang, Zixiang Li, LiPing Zhang, and Chaoyong Zhang. 2017. Balancing stochastic two-sided assembly line with multiple constraints using hybrid teaching-learning-based optimization algorithm. Comput. Oper. Res. 82, C (June 2017), 102–113. DOI:https://doi.org/10.1016/j.cor.2017.01.015