

DESIGN OF A NOVEL HYBRID HEURISTICS ALGORITHM TO SIMULATE THE SEQUENCE DEPENDENT SETUPS WITH BACKLOGGING FOR MFSP

DISSERTATION II

*Submitted in partial fulfillment of the
requirement for the award of the
Degree of*

MASTER OF TECHNOLOGY
IN
Manufacturing Engineering

By

*Deep Singh Malhi
(Reg. No. 11206161)*

Under the Guidance of

Harpreet Singh
(Assistant Professor)



PHAGWARA (DISTT. KAPURTHALA), PUNJAB

(School of Mechanical and Industrial Engineering)
Lovely Professional University
Punjab
May 2017

**DESIGN OF A NOVEL HYBRID HEURISTICS ALGORITHM
TO SIMULATE THE SEQUENCE DEPENDENT SETUPS
WITH BACKLOGGING FOR MFSP**

DISSERTATION II

*Submitted in partial fulfillment of the
requirement for the award of the
Degree of*

**MASTER OF TECHNOLOGY
IN
Manufacturing Engineering**

By

Deep Singh Malhi
(Reg. No. 11206161)
Under the Guidance of

Harpreet Singh
(Assistant Professor)



(School of Mechanical and Industrial Engineering)
Lovely Professional University
Punjab
May 2017



Lovely Professional University Jalandhar, Punjab

CERTIFICATE

I, hereby certify that the work which is being presented in the dissertation entitled **“Design of a Novel Hybrid Heuristics Algorithm to Simulate the Sequence Dependent Setups with Backlogging for MFSP”** in partial fulfillment of the requirement for the award of degree of **Master of Technology (Manufacturing Engineering)** and submitted in Department of Mechanical Engineering, Lovely Professional University, Punjab is an authentic record of my own work carried out during period of Dissertation under the supervision of **Mr. Harpreet Singh, Assistant Professor**, Department of Mechanical Engineering, Lovely Professional University, Punjab.

The matter presented in this dissertation has not been submitted by me anywhere for the award of any other degree or to any other institute.

Date: **(Deep Singh Malhi)**
Reg. No. 11206161

This is to certify that the above statement made by the candidate is correct to best of my knowledge.

Date: **(Harpreet Singh)**
Supervisor

Date: COD (ME)

The M- Tech Dissertation II examination of Deep Singh Malhi, has been held on

_____.

Signature of Examiner



**LOVELY PROFESSIONAL UNIVERSITY
PHAGWARA, PB, INDIA**

CANDIDATE'S DECLARATION

I, Deep Singh Malhi, Reg. No. 11206161 hereby declare that the work presented entitled **“DESIGN OF A NOVEL HYBRID HEURISTICS ALGORITHM TO SIMULATE THE SEQUENCE DEPENDENT SETUPS WITH BACKLOGGING FOR MFSP”** in partial fulfillment of requirements for the award of Degree of Master of Technology (Manufacturing Engineering) submitted to the Department of Mechanical Engineering at Lovely Professional University, Phagwara is an authentic record of my own work carried out during the period from AUG 2016 to MAY 2017, under the supervision of Mr. Harpreet Singh (Assistant Professor), Department of Mechanical Engineering. The matter presented in this thesis has not been submitted in any other University/ Institute for the award of Degree of Master of Technology. Furthermore, I also declare that I will not publish this work in any other Journals/ Conferences/ Workshop seminars except the one chosen by supervisor. The presented work is the property of Lovely Professional University, Phagwara. If I found violating any of the above conditions, University has right to cancel my degree.

Signature of Student

Date:

Place:

ABSTRACT

The present research report addresses the formulation of novel hybrid heuristics algorithm for multi-objective flow shop scheduling problems with dual objectives of minimizing the process time and completion time, constrained by sequence dependent setups and backlogging. The purpose of present work is to develop a model using two heuristics, capable of minimizing the considered objectives and hence leads to increase in capacity utilization efficiency thereby reducing the time required to complete jobs, and consequently increasing the profitability of an organization in present competitive environment. The two adopted heuristics are Teacher Learning based Optimization (TLBO) and Biogeography Based Optimization (BBO). TLBO utilizes the concept of interaction between teacher and learners, and then among learners themselves to increase knowledge level. On the other hand, BBO is based on natural process of migration and emigration of the species from one habitat to other. A novel hybrid heuristics algorithm namely BBO-TLBO is proposed, which includes teacher phase of TLBO between the migration and mutation phase of BBO and is implemented to assess a manufacturing system of two products-eight machines. Multi-objective scheduling with SDST also becomes NP hard with greater complexity towards the optimality in a reasonable time. Hence, heuristics has become greater choice for solving NP hard problems because of their multi-solution and strong neighborhood search capabilities in a reasonable time. MATLAB is utilized to evaluate the different functions of both objectives. The results using proposed BBO-TLBO are compared with the present industrial outputs and respective validation is conducted case study wise. The proposed work formulates eight different functions for both the objectives, respectively. The post results generated proved to be better than actual data. For objective of minimization of process time, the comparative and optimized mean values (in minutes) for both the parts on machine M1 is 9.233; M2 is 6.209; M3 is 10.619; M4 is 23.976; M5 is 11.685; M6 is 11.685; M7 is 22.044 and M8 is 14.364. For objective of minimization of cycle time, the comparative and optimized mean values (in minutes) for both the products on machine M1 is 8.755; M2 is 12.405; M3 is 12.975; M4 is 32.442; M5 is 8.838; M6 is 22.372; M7 is 38.453 and M8 is 32.886. In this regard, it is determined from the analytical results that examined time estimates are within range of lower and upper bounded limits of cycle time of 3.8 to 10.3 minutes for part 1 and from 4.8 to 12.2 minutes for part 2. Simultaneously, the process time for part 1 ranges between 1.4 to 11.3 minutes and from 4.8 to 12.2 minutes

for part 2. Therefore, the proposed model finds as validated and model can be implemented in industry to minimize the process time and cycle time, hence the manufacturing capacity of the system. Also, the reduced time refers the effectiveness of the proposed hybrid heuristics algorithm and hence, it makes the system more reliable. The diversity in the optimization values shows the effectiveness of the proposed hybrid heuristic. Number of iterations taken for evaluation of every function are 20 and it takes 3 seconds to evaluate each function on 1.5 GHz quad core processor.

ACKNOWLEDGEMENT

Nothing can be achieved without an optimal combination of inspiration and perspiration. The real spirit of achieving a goal is through the way of excellence and discipline. I would have never succeeded in completing my task without the cooperation, encouragement and help provided to me by various personalities.

I owe my special thanks to Mr. Harpreet Singh, Assistant Professor, Department of Mechanical Engineering, Lovely Professional University, Phagwara, who helped and guided me for this work. His encouraging remarks from time to time greatly helped me in improving my skills.

I am also thankful to Mr. Gurpreet Singh Phull, HOS, Department of Mechanical Engineering, Lovely Professional University, Phagwara. I wish to place on record my gratitude for all those who have been instrument in bringing my work to this stage.

I do not find enough words with which I can express my feeling of thanks to all my family members and friends for their help, inspiration and moral support which went a long way in completion of my thesis.

Above all I render my gratitude to the ALMIGHTY and MY PARENTS who bestowed me self-confidence, ability and strength to complete this work.

Deep Singh Malhi
Reg. No. 11206161

LIST OF FIGURES

Figure No.	Description	Page No.
Fig. 1.1	Depiction of Classification Scheduling Problem	02
Fig. 1.2	Depictions of Various Methods to Solve Scheduling Problems	07
Fig. 1.3	Flow Process Chart Depicting Steps in GA for Optimization	08
Fig. 3.1	Depiction of Process of TLBO for Optimization of MFSP	46
Fig. 3.2	Depiction of Relation between Emigrations, Immigration Rates w.r.t. Number of Species for BBO Algorithm	48
Fig. 3.3	Depiction of Various Steps Included in BBO for Optimization of MFSP	49
Fig. 3.4	Depiction of the Step 1 which Defines the Problem in MATLAB Software	51
Fig. 3.5	Depiction of Step 2, which Defines the Parameters in MATLAB Software	52
Fig. 3.6	Depiction of Step 3 which Initializes the Algorithm in MATLAB Software	53
Fig. 3.7	Depiction of Teacher Phase in Proposed Heuristic in MATLAB Software	55
Fig. 3.8	Depiction of Various Steps of Proposed Hybrid Heuristic Optimization of MFSP	56
Fig. 3.9	Depiction of the Flow Process of Products in Piston Manufacturing Industry	58
Fig. 4.1	Implementation of Proposed BBO-TLBO Heuristic for MFSP	63
Fig. 4.2	Graph Showing the Variation of Function $f=11x(1)+12x(2)\leq 60$ Corresponding to the Number of Iterations	65
Fig. 4.3	Graph Showing the Variation of Function $f=11x(1)+11x(2)\leq 90$ Corresponding to the Number of Iterations	67
Fig. 4.4	Graph Showing the Variation of Function $f=12x(1)+15x(2)\leq 90$ Corresponding to the Number of Iterations	68
Fig. 4.5	Graph Showing the Variation of Function $f=19x(1)+21x(2)\leq 100$ Corresponding to the Number of Iterations	70
Fig. 4.6	Graph Showing the Variation of Function $f=11x(1)+13x(2)\leq 60$ Corresponding to the Number of Iterations	71

Fig. 4.7	Graph Showing the Variation of Function $f=15x(1)+14x(2)\leq 120$ Corresponding to the Number of Iterations	73
Fig. 4.8	Graph Showing the Variation of Function $f=19x(1)+24x(2)\leq 90$ Corresponding to the Number of Iterations	74
Fig. 4.9	Graph Showing the Variation of Function $f=15x(1)+17x(2)\leq 120$ Corresponding to the Number of Iterations	76
Fig. 4.10	Graph Showing the Variation of Function $f=17x(1)+18.5x(2)\leq 60$ Corresponding to the Number of Iterations	77
Fig. 4.11	Graph Showing the Variation of Function $f=18x(1)+18x(2)\leq 90$ Corresponding to the Number of Iterations	79
Fig. 4.12	Graph Showing the Variation of Function $f=19.5x(1)+23.5x(2)\leq 90$ Corresponding to the Number of Iterations	80
Fig. 4.13	Graph Showing the Variation of Function $f=30.5x(1)+34x(2)\leq 100$ Corresponding to the Number of Iterations	82
Fig. 4.14	Graph Showing the Variation of Function $f=18x(1)+19.5x(2)\leq 60$ Corresponding to the Number of Iterations	83
Fig. 4.15	Graph Showing the Variation of Function $f=24x(1)+22x(2)\leq 120$ Corresponding to the Number of Iterations	85
Fig. 4.16	Graph Showing the Variation of Function $f=33x(1)+34.5x(2)\leq 100$ Corresponding to the Number of Iterations	86
Fig. 4.17	Graph Showing the Variation of Function $f=31.5x(1)+30x(2)\leq 120$ Corresponding to the Number of Iterations	87
Fig. 4.18	Depiction of Set up Time, Process Time, Backlog Time, Cycle Time and Reduced Cycle Time for Part 1	89
Fig. 4.19	Depiction of Set up Time, Process Time, Backlog Time, Cycle Time and Reduced Cycle Time for Part 2	90
Fig. 4.20	Graph Displaying the Actual Process time and Results by BBO-TLBO Heuristic	93
Fig. 4.21	Graph Showing Results of BBO-TLBO Heuristic and Optimized Time (in min)	93
Fig. 4.22	Graph Displaying the Actual Process Time and Optimized Time (in min)	94
Fig. 4.23	Graph Showing the Cumulative Validation of Actual data, results by BBO-TLBO and Optimized Data for Part 1	95

Fig. 4.24	Graph Showing the Actual Process Time and Results by BBO-TLBO Heuristic (in min)	96
Fig. 4.25	Graph Showing Results by BBO-TLBO Heuristic and Optimized Process Time (in min)	97
Fig. 4.26	Graph Showing Actual Process Time and Optimized Process Time (in min)	98
Fig. 4.27	Graph Showing the Cumulative Validation of Actual Data, Results by BBO-TLBO and Optimized Data for Part 2	99
Fig. 4.28	Graph Displaying the Actual Data of Cycle Time and Results Generated (in minutes) from Proposed BBO-TLBO Heuristic for Part 1	100
Fig. 4.29	Graph Showing the Results Generated (in minutes) from Proposed BBO-TLBO Heuristic and the Optimized Data for Part 1	101
Fig. 4.30	Graph Showing Actual Data and Optimized Data for Part 1	102
Fig. 4.31	Graph Showing the Cumulative Display of Actual Data, Results Generated from Proposed BBO-TLBO Heuristic and Optimized Data (in minutes) for Part 1	103
Fig. 4.32	Graph Displaying the Actual Data of cycle Time and Results Generated (in minutes) from Proposed BBO-TLBO Heuristic for Part 2	104
Fig. 4.33	Graph showing the Results Generated (in minutes) from Proposed BBO-TLBO Heuristic and the Optimized Data for Part 2	104
Fig. 4.34	Graph Showing the Actual Cycle Time Data and Optimized Cycle Time for Part 2	105
Fig. 4.35	Graph Showing the Cumulative Display of Actual Data, Results Generated from Proposed BBO-TLBO Heuristic and Optimized Data (in minutes) for Part 2	106
Fig. 4.36	Graph Depicting the Total Process Time and Total Optimized Time for Part 1 and Part 2 on Various Machines	108
Fig. 4.37	Graph Depicting the Total Cycle Time and Total Optimized Cycle Time for Part 1 and Part 2 on Various Machines	109

LIST OF TABLES

Table No.	Description	Page No.
Tab. 3.1	Model Problem of Flow Shop Scheduling with 2 Parts and 8 Machines	44
Tab. 3.2	Depiction of the Different Times of Part 1 on Various Machines	59
Tab. 3.3	Depiction of the Different Times of Part 2 on Various Machines	60
Tab. 4.1	Evaluation and Optimization of Function $11x(1)+12x(2) \leq 60$ Using Proposed BBO- TLBO Heuristic	65
Tab. 4.2	Evaluation and Optimization of Function $11x(1)+11x(2) \leq 90$ Using Proposed BBO- TLBO Heuristic	66
Tab. 4.3	Evaluation and Optimization of Function $12x(1)+15x(2) \leq 90$ Using Proposed BBO- TLBO Heuristic	68
Tab. 4.4	Evaluation and Optimization of Function $19x(1)+21x(2) \leq 100$ Using Proposed BBO- TLBO Heuristic	69
Tab. 4.5	Evaluation and Optimization of Function $11x(1)+13x(2) \leq 60$ Using Proposed BBO- TLBO Heuristic	71
Tab. 4.6	Evaluation and Optimization of Function $15x(1)+14x(2) \leq 120$ Using Proposed BBO- TLBO Heuristic	72
Tab. 4.7	Evaluation and Optimization of Function $19x(1)+24x(2) \leq 90$ Using Proposed BBO- TLBO Heuristic	74
Tab. 4.8	Evaluation and Optimization of Function $15x(1)+17x(2) \leq 120$ Using Proposed BBO- TLBO Heuristic	75
Tab. 4.9	Evaluation and Optimization of Function $17x(1)+18.5x(2) \leq 60$ Using Proposed BBO- TLBO Heuristic	77
Tab. 4.10	Evaluation and Optimization of Function $18x(1)+18x(2) \leq 90$ Using Proposed BBO- TLBO Heuristic	78
Tab. 4.11	Evaluation and Optimization of Function $19.5x(1)+23.5x(2) \leq 90$ Using Proposed BBO- TLBO Heuristic	80
Tab. 4.12	Evaluation and Optimization of Function $30.5x(1)+34x(2) \leq 100$ Using Proposed BBO- TLBO Heuristic	81
Tab. 4.13	Evaluation and Optimization of Function $18x(1)+19.5x(2) \leq 60$ Using Proposed BBO- TLBO Heuristic	83
Tab. 4.14	Evaluation and Optimization of Function $24x(1)+22x(2) \leq 120$ Using Proposed BBO- TLBO Heuristic	84

Tab. 4.15	Evaluation and Optimization of Function $33x(1)+34.5x(2)\leq 100$ Using Proposed BBO- TLBO Heuristic	85
Tab. 4.16	Evaluation and Optimization of Function $31.5x(1)+30x(2)\leq 120$ Using Proposed BBO- TLBO Heuristic	87
Tab. 4.17	Depiction of the Various Times (in min) of Part 1 on Various Machines	89
Tab. 4.18	Depiction of the Various Times of Part 2 on Various Machines	90
Tab. 4.19	Depiction of the Actual Data of Process Time (in minutes) from Industry	92
Tab. 4.20	Depiction of Results Generated (in minutes) from Proposed BBO- TLBO Heuristic	92
Tab. 4.21	Depiction of Actual Process Time and Results by BBO-TLBO Heuristic (in min)	92
Tab. 4.22	Depiction of Results by BBO-TLBO Heuristic and Optimized Time (in min)	93
Tab. 4.23	Depiction of Actual Process Time and Optimized Time (in min)	94
Tab. 4.24	Depiction of Cumulative Validation of Actual Data, Results by BBO-TLBO and Optimized Data for Part 1	95
Tab. 4.25	Depiction of Actual Process Time and Results by BBO-TLBO Heuristic (in min)	96
Tab. 4.26	Depiction of Results by BBO-TLBO Heuristic and Optimized Time (in min)	96
Tab. 4.27	Depiction of Actual Process Time and Optimized time (in min)	97
Tab. 4.28	Depiction of Cumulative Validation of Actual data, Results by BBO-TLBO and Optimized Data for Part 2	98
Tab. 4.29	Depiction of the Actual Data of Cycle Time (in minutes) from Industry	99
Tab. 4.30	Depiction of the Results Generated (in minutes) from Proposed BBO-TLBO Heuristic	99
Tab. 4.31	Depiction of Actual Data of cycle Time and Results Generated (in minutes) from Proposed BBO-TLBO Heuristic for Part 1	100
Tab. 4.32	Depiction of Results Generated (in minutes) from Proposed BBO- TLBO Heuristic and the Optimized Data for Part 1	100
Tab. 4.33	Depiction of Actual Data and Optimized Data for Part 1	101

Tab. 4.34	Depiction of Actual Data, Results Generated from Proposed BBO-TLBO Heuristic and Optimized Data (in minutes) for Part	102
Tab. 4.35	Depiction of Actual Data of cycle Time and Results Generated (in minutes) from Proposed BBO-TLBO Heuristic for Part 2	103
Tab. 4.36	Depiction of Results Generated (in minutes) from Proposed BBO-TLBO heuristic and the Optimized Data for Part 2	104
Tab. 4.37	Depiction of Actual Data and Optimized Data for Part 2	105
Tab. 4.38	Depiction of Actual Data, Results Generated from Proposed BBO-TLBO Heuristic and Optimized Data (in minutes) for Part 2	106
Tab. 4.39	Cumulative Depiction of Total Process Time and Total Optimized Time for Part 1 and Part 2 on Various Machines	107
Tab. 4.40	Cumulative Depiction of Total Cycle Time and Total Optimized Time for Part 1 and Part 2 on Various Machines	109

NOMENCLATURE

ABBO	Adaptive Biogeography- Based Optimization
ACO	Ant Colony Optimization
AIS	Artificial Immune Algorithm
AMH	Akartunali and Miller Heuristic
ANOVA	Analysis of Variance
APNA	Advanced Non Population Based Algorithm
ARB	Attributed Ratio of Batch
BBO	Biogeography- Based Optimization
B&B	Branch and bound
B&C	Branch-and-Cut
BO	Bacterial Optimization
BOH	Bi-Objective Heuristic
BOLS	Bi-Objective Local Search Algorithm
BSA	Backtracking Search Algorithm
CFSMP	Constrained Flow Shop with Multiple Processes Problem
CVRP	Capacitated Vehicle Routing Problem
DABC	Discrete Artificial Bee Colony
DAPFSP	Distributed Assembly Permutation Flow-Shop Scheduling Problem
DE	Differential Evolution
DOE	Design of Experiments
DTSAFSP	Distributed Two-Stage Assembly Flow-Shop Problem
DVNS	Dynamic Variable Neighborhood Search
EDD	Earliest Due Date
EOX	Enclosed Order Crossover
FO	Fix and Optimize
FJSRA	Fictitious Job Setup Ranking Algorithm
FPSO	Farness Particle Swarm Optimization
GA	Genetic Algorithm
GVNS	General Variable Neighborhood Search
HBBO	Hybrid Biogeography Optimization Algorithm
HBSA	Hybrid Backtracking Search Algorithm

HDDE-RVNS	Hybrid Discrete Differential Evolution-Reduced Variable Neighborhood Search
HFSPB	Hybrid Flow Shop Parallel Batching
HFSMT	Hybrid Flow Shop Scheduling with Multiprocessor Task
HGA	Hybrid Genetic Algorithm
HGA-RVNS	Hybrid Genetic Algorithm-Reduced Variable Neighborhood Search
HMOIA	Hybrid Multi-Objective Immune Algorithm
HMPGA	Hybrid Multi Population Genetic Algorithm
HPDE	Hybrid Permutation Based Differential Evolution
HSI	High Suitable Index
HSMO	Harmony Search Based Multi-Objective Optimization Model
HVNS	Hybrid Variable Neighborhood Search
IA	Immune Algorithm
ICA	Imperialistic Competitive Algorithm
IEA	Immune Evolutionary Algorithm
IIS	Individual Improving Scheme
IG	Iterated Greedy
IS	Immune System
ISP	Iterated Swap Procedure
JIT	Just in Time Production
LFEPSO	Levy Flight Embedded Particle Swarm Optimization
LOV	Largest Order Value
LSL	Least Slack Time
MACS	Multi-Ant Colony System
MARB	Minimum Attribute Ratio of Batch
MFL	More-for Less Solution
MFSP	Multi-Objective Flow Shop Scheduling
MILP	Mixed Integer Linear Program
MINLP	Mixed Integer Non-Linear Program
MIP	Mix-Integer Linear Programming Model
MMOP	Multi-objective Optimization Problem
MMPPRP-TW	Multi-Vehicle Production and Pollution Routing Problems with a Time Window

MOACSA	Multi-Objective Ant-Colony System Algorithm
MODE	Multi-objective Differential Evolutionary MOPSO Multi-Objective Particle Swarm Optimization
MOSA	Multi-Objective Simulated Annealing
MOSLS	Multi-Objective Stochastic Local Search
MPGA	Multi Population Genetic Algorithm
NEH_RMB	Nawaz Enscore Ham and Rio Mercado and Brad Heuristic
NRGA	Non-Dominated Ranked Genetic Algorithm
NSGA	Non-Dominated Sorting Genetic Algorithm
ORPF	Optimal reactive Power Flow
OSL	Overall Slack Time
PCB	Printed-Circuit-Board
PEM	Prediction Error Method
PFSP	Permutation Flow Shop Scheduling Problem
PH	Polynomial Time Heuristic
PR	Path Relinking
PSO	Particle Swarm Optimization
PTS	Parallel Tabu Search
RCPSPDC	Resource Constrained Project Scheduling Problem with Discounted Cash Flows
RFL	Referred Local Search
RIPG	Restarted Iterated Pareto Greedy
RL	Repetitive Lots Procedure
RIP	Resource Investment Problem
RLP	Resource Leveling Problem
SA	Simulated Annealing
SAA	Simulated Annealing Algorithm
SAL	Sided Assembly Line
SDST	Sequence Dependent Setup Times
SGA	Smart Decoding-Based Genetic Algorithm
SLPSO	Self-Learning Particle Swarm Optimization
SPT	Shortest Processing Time
SRA	Setup Ranking Algorithm
SS	Scatter Search

TG	Two-group Heuristic
TLBO	Teacher Learning Based Optimization
TLH	Total Labor Hours
TOPSIS	Technique for Order Preference By Similarity To Ideal Solution
TPSGA	Two Phase Subpopulation Genetic Algorithm
TRL	Truncated Repetitive Lots Procedure
TS	Tabu Search
TSAFSP	Two-Stage Assembly Flow-Shop Scheduling Problem
TTT	Total Throughput Time
UACOR	Unified Ant Colony Optimization Algorithm
VESA	Vector Evaluated Simulated Annealing
VND	Variable Neighborhood Descent
VNS	Variable Neighborhood Search
ZWSP	Zero Wait Scheduling Problems
K	Setup Penalization Parameter
PT	Processing Time
TS	Setup Time
[C]	Minimize Cycle Time
[Z]	Minimize Process Time
C_j	Completion Time, where j represents job
Difference _D	Difference between two means
D_n	Number of variables
E_j	Earliness, where j represents job
G_n	Generations to be computed
J_{MN}	Part M is processing on machine N
Liv	Workload leveling function
L_l	Lower bound
M_D	Mean value of population
P_n	Population size
T_F	Teaching factor
T_j	Tardiness, where j represents job
U_l	Upper bound
X_i	Knowledge level before interaction

X_j	Knowledge level after interaction
X_{new}	New updated value of learner
X_{old}	Old knowledge value of learner
X_{teacher}	Knowledge value of teacher
$X(n)$	Variable representing time
j_z	Job, where z represents number of job
m_i	Machine and i represents number of machine
w_1, w_2	Weight given to earliness and tardiness respectively

CONTENTS

Certificate	i
Declaration	ii
Abstract	iii
Acknowledgement	v
List of Figures	vi
List of Tables	ix
Nomenclature	xii
Contents	xvii

CHAPTER 1: INTRODUCTION

1.1 Scheduling	01
1.1.1 General Terms Describing a Job in a Scheduling Problem	01
1.1.2 Assumptions in Scheduling	02
1.1.3 Classification of Scheduling	02
1.2 Heuristics Approaches	06
1.2.1 Different Types of Heuristics to Solve Scheduling Problems	07
1.2.2 Genetic Algorithms	07
1.2.3 Simulated Annealing	09
1.2.4 Branch and Bound Method	11
1.2.5 Ant Colony Optimization	12
1.2.6 Particle Swarm Optimization	14
1.2.7 Tabu Search	15
1.2.8 Differential Evolution	16
1.2.9 Immune Algorithm	18
1.2.10 Teacher Learning Based Optimization (TLBO)	19
1.2.11 Biogeography Based Optimization (BBO)	20
1.3 Multi-Objective Flow Shop Scheduling Problems (MFSP)	21
1.4 Sequence Dependent Setup Time (SDST)	22
1.5 Backlogging in Scheduling	23

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction	25
2.2 Research Models	25

CHAPTER 3: PRESENT WORK

3.1 Introduction	40
3.2 Problem Formulation	41
3.2.1 Assumptions	42
3.2.2 Considerations	42
3.3 Research Gaps	42
3.4 Objectives	43
3.5 Objective Function	44
3.6 Heuristic Methods and Hybrid Algorithm	45
3.6.1 Teacher Learning Based Optimization (TLBO)	45
3.6.2 Biogeography Based Optimization (BBO)	47
3.6.3 Novel Hybrid Heuristic: TLBO's Teacher Phase Based BBO (BBO-TLBO)	50
3.7 Collection of Data from Industry to Solve MFSP	57
3.7.1 Process Flow-Piston Machinery Shop	57
3.7.2 Business Plan of Piston Manufacturing Industry	57
3.7.3 Database Sheet	58
3.7.4 Data Analysis and Documentation	59
3.7.5 Formation of the Objective Functions for Part 1 and Part 2	60
3.8 Description of Software Used to Implement the Novel Hybrid BBO- TLBO Heuristic for MFSP	62
CHAPTER 4: RESULTS AND DISCUSSIONS	
4.1 Implementation of Proposed Hybrid BBO-TLBO Heuristic for MFSP	63
4.2 Mathematical Analysis Using Proposed BBO-TLBO for Optimization of Process Time and Cycle Time for MFSP	64
4.2.1 Mathematical Analysis for Minimization of Process Time Using the Proposed BBO-TLBO Heuristic	64
4.2.2 Mathematical Analysis for Minimization of Cycle Time Using the Proposed Hybrid BBO-TLBO Heuristic	76
4.3 Simulation of Sequence Dependent Setup Time Along With Backlogging for Part 1 and Part 2	88
4.3.1 Simulation of Set up and Backlogging for Part 1	88
4.3.2 Simulation of Set up time and Backlogging for Part 2	90
4.4 Validation of Results Generated by Proposed BBO-TLBO Heuristic for Minimization of Process Time and Cycle Time	91

4.4.1 Validation of Results for the Objective of Minimization of Process Time	92
4.4.2 Validation of Results for the Objective of Minimization of Cycle Time	99
4.4.3 Optimization of the Model	107
4.5 Discussion of Results	110
4.5.1 Discussion of Results for Minimization of Process Time	110
4.5.2 Discussion of Results for Minimization of Cycle Time	111
CHAPTER 5: CONCLUSIONS AND FUTURE SCOPE OF WORK	
5.1 Conclusions	113
5.2 Recommendations	115
5.3 Future Scope of Work	116
REFERENCES	118

CHAPTER 1: INTRODUCTION

1.1 SCHEDULING

In the real time scenario, there exist many situations in manufacturing system like due date changes, unexpected job release, machine breakdowns and greater processing times, than estimated and expected. The cost of production aggregates to high proportion of any firm's expenditure, hence every firm tries to get a proper design of shop and scheduling of jobs on various machines to optimize the task times for long-term and short-term goals. Scheduling, hence, is a non-ignorable aspect of every manufacturing system. Scheduling is the allocation of limited resources (man and machinery), by organizing, controlling and optimizing various set of activities in a manufacturing process in a specific amount of time. Simply, it authenticates the production facility when to make, on which equipment and with which staff. Johnson (1954) studied two and three stage production system with included set-up times. Li and Willis (1992) mentioned two aspects, forward scheduling, in which activities are planned from the date they become available to determine the due date, and backward scheduling, in which activities are planned for later as possible, to meet the due date. Singh and Mahapatra (2012) and Huang et al. (2014) specified the various criteria as minimizing the make-span, lateness, machine idle cost, inventory cost and tardiness. The objective is to increase the production efficiency, optimization of resources, minimizing production cost and increase in competitive strength. Rossi (2016) defined it as a set of 'n' jobs (different or same) processed on 'm' set of machines to minimize the given criterion. Scheduling is an effective method to plan the sequence of tasks and is applicable to service industry, project control, electronic industry, computer science, food processing industry, chemical, textile and so on.

1.1.1 General Terms Describing a Job in a Scheduling Problem

The following terms describe jobs in single machine scheduling problem.

- (i) Processing time: It is the time required to process job 'j'. It includes both, the actual processing time and set-up time.
- (ii) Ready time: It is the difference between the arrival time of the job and the time at which, the processing of job starts.
- (iii) Due date: It is the time at which the processing of the job j is to be completed.
- (iv) Completion Time: It is the time at which the job 'j' is actually completed in a sequence.

(v) Flow time: It is the amount of time that job 'j' spends in a system. It is difference between completion and ready time.

(vi) Lateness: It is the amount of time by which completion time of job 'j' differs from its due date. It can be positive or negative. Positive lateness implies completion of job after its due date, and is a measure of poor service, while negative lateness is measure of better service.

(vii) Tardiness: it is the lateness of job 'j', if it fails to meet its due date, else it will be zero. The maximum of zero and difference of completion and due date is tardiness.

1.1.2 Assumptions in Scheduling

The following conditions prescribe the scheduling process:

1. A set of 'n' independent jobs, each with single operation is available for processing at time zero.
2. Set-up time of each of the jobs is independent of its position in the sequence of jobs. Therefore, the set-up tie of each job can be included in its processing time.
3. Job descriptors are to know in advance.
4. One machine is continuously available and is never kept idle when work waiting.
5. Each job is processed until its completion without any pre-emption.

1.1.3 Classification of Scheduling

Scheduling is allocating the resources from the initial to the final times, for the various tasks associated with different jobs, to optimize some performance measures. Figure 1.1 depicts classification of scheduling problem.

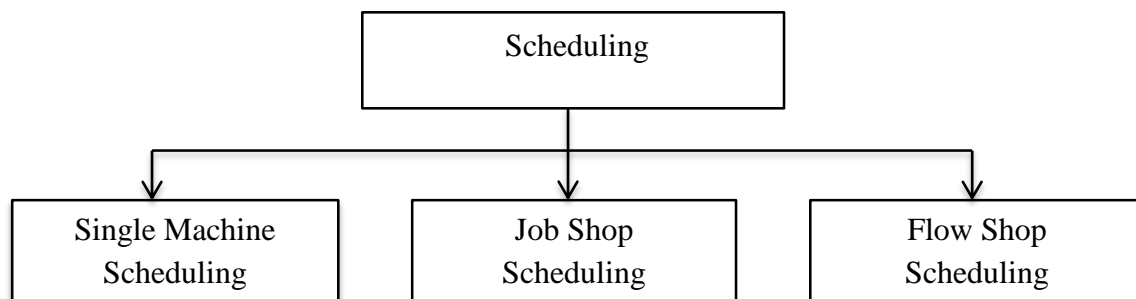


Figure 1.1: Depiction of Classification Scheduling Problem

(i) Single machine scheduling

Single Machine scheduling, consists of 'n' independent jobs, each with single operation. The objective of minimizing the make span is achieved by arranging the jobs on the basis of in-

process time, implying that, jobs with less in-process time are put ahead of those with higher in-process times.

(ii) Job Shop Scheduling

In job shop scheduling, each job have 'm' different operations. If jobs are having less than m operations, we assume required number of dummy operations with zero process times, ensuring the condition of equal number of operations. Graham (1966) first presented the competitive analysis for such combinatorial optimization problem. Taillard (1993) discussed the various problems for basic model to minimize the make-span. The process sequences of jobs are not same. Hence, the flow of each job is not unidirectional and it is not compulsory to process all the jobs on each machine available that means each distinct job may be processed in any distinct number of machines as requirement demands. Neither, an initial machine is there to perform only first operation of a job, nor any terminal machine to perform only last operation of job. Each operation 'j' in operation sequence of job I in the job shop problem is described as triplet (i, j, k), where 'k' is the required machine for processing the jth operation of ith job.

(iii) Flow shop scheduling

In flow shop environment, each job has to go through a series of operations in the same order, implying that the jobs have to follow the same route or process sequence, but the processing time of each operation on a job will be different from that of other jobs. Garey et al. (1976) studied the complexity in the flow shop environment. While evaluating a flow shop problem, we examine all possible sequences before carrying out the job, and then the best get chosen among those. A flow-shop scheduling problem involves scheduling 'n' jobs on 'm' machines. A job consists of 'm' tasks and the jth operation of each job must be process on the jth machine, which only happens if it has already completed on machine $j - 1$ and machine j is idle. The time taken to process a job on a machine may be constant, non-negative or even zero. In the case where the processing time of a job on a machine is zero, implying that the job is not processed on the machine. Graham et al. (1979) studied sequencing and scheduling w.r.t optimization and approximation algorithms and interpreted in computational complexity theory. Potts (1980) presented a branch and bound algorithm for permutation flow-shop problem, which included dominance rules, and, computed upper bounds to minimize the maximum total flow time. Bellman et al. (1982) discussed two-

machine flow shop scheduling problem with an aim to find a job sequence to minimize the make span, in which, one machine was assumed to had sequence dependent set-up times. Graves et al. (1983) developed a computerized heuristic procedure for the scheduling of the re-entrant flow shop environment including photolithographic process applicable to flexible machining system and integrated circuit fabrication process. Hariri et al. (1984) considered two-machine flow shop problem with constraints for minimizing the completion time, and developed three branch and bound algorithms including lower bounds, developed using Lagrangian relaxation. Scudder and Hoffmann (1985) investigated the value based dispatching rules for flow shop environment to obtain on-time performance simultaneously reducing the investment in inventory Boxma and Forst (1986) considered stochastic scheduling problems involving both single machine and arbitrary number of machines with an objective to minimize the weighted number of tardy jobs when all the jobs have due dates. Flynn et al. (1987) investigated Repetitive Lots Procedure (RL) and Truncated Repetitive Lots (TRL) procedure, which uses sequence dependent set-up times to improve the performance of group technology manufacturing. Nowicki et al. (1988) proposed two heuristics for two-machine flow shop problem whose decision variables are sequence of jobs and processing time on which cost of performing a job depends linearly. Potts et al. (1990) proposed a heuristic method that aim at minimizing the length of flow shop schedule when lot streaming is in action. Hunsucker and Shah (1992) developed a simulation model for constrained flow shop with multiple processes problem (CFSMP) considering mean tardiness and number of tardy jobs as performance measures. Sarin et al. (1993) addressed a heuristic to minimize idle time on last machine and determining the sequence to prioritize the jobs to minimize total completion time. Halim et al. (1994) proposed the optimal algorithms to solve the problem of multi due-date, determining and scheduling the batch size in order to minimize actual flow time in JIT environment. Uetake et al. (1995) experimented two-stage hybrid flow- shop problem consisting of one machine at first stage and several kinds at second stage, to schedule the jobs taking make-span and maximum work-in process as performance measures. Kawtummachai (1997) applied genetic algorithm and simulated annealing to automated flow shop environment to minimize the total cost of scheduling of orders during production. Ben-Daya and Al-Fawzan (1998) proposed tabu search for permutation flow shop problem in combination with collaborated scheme for intensification

and diversification, for generating neighborhood of given sequence. Botta-Genoulaz et al. (2000) proposed six heuristics to minimize maximum lateness in hybrid flow shop scheduling which considers precedence constraints, time lags and due-dates. Ouenniche and Boctor (2001) presented a Two-group Heuristic (TG), to solve multi-product, scheduling and economic lot sizing problem to minimize work-in-progress inventory, holding costs, setup costs without backlogging. Jain and Meeran (2002) applied Scatter Search (SS) and Path Relinking (PR) to explore the search space for the best solutions in the flow shop environment, which allows passing during the processing. Gourgand et al. (2003) proposed a recursive algorithm based on Markov chain and a simulated model for a stochastic flow-shop scheduling problem in order to minimize the make-span. Nearchou (2004) presented a hybrid Simulated Annealing Algorithm (SAA), consisting the features of local search technique and GAs to improve the performance for flow shop problem. Kumar et al. (2006) proposed a psycho-clonal algorithm based on artificial immune system and Maslow's need hierarchy theory for no-wait flow shop problem to reduce total flow time. Moghaddam et al. (2007) investigated a Hybrid Multi-Objective Immune Algorithm (HMOIA) for minimizing weighted mean time and weighted mean tardiness using Immune System (IS) and Bacterial Optimization (BO) to find Pareto solution. Xu et al. (2008) presented a model to minimize one of the three criteria's, namely, total weighted completion time, discounted total weighted completion time and sum of quadratic job completion time by using optimal permutations and formed an algorithm. Moslehi et al. (2009) introduced an algorithm using Branch and Bound (B & B) method with upper and lower bounds with an objective to minimize sum of maximum earliness and tardiness, useful in Just in Time production (JIT) system. Kahraman et al. (2010) formulated a greedy algorithm with two phases, destructive and constructive, along with four constructive heuristics to solve Hybrid Flow Shop Scheduling with Multiprocessor Task (HFSMT) to minimize the total completion time. Pan et al. (2011), proposed a Discrete Artificial Bee Colony (DABC), considering total weighted earliness and tardiness penalties as parameters to solve lot-streaming flow shop scheduling problem under both idling and no-idling cases. The rules smallest overall slack time (OSL), Earliest Due Date (EDD) and the smallest slack time on the last machine (LSL) was employed to construct diversity and quality in population. Ahmadizar et al. (2012) proposed a hybrid genetic algorithm for open shop scheduling to minimize the make-span. The algorithm uses

mutation operator, to prevent searching of repetitive solutions, and crossover operator for preserving the order of jobs relatively on machines. Based on three concepts, lower bound to reduce search space, dispatching index supported on longest remaining process time and randomized active schedules, and employed an iterative heuristic as an optimized measure. Mousavi et al. (2013) introduced BOLS i.e. bi-objective local search algorithm for hybrid flow shop with bi-objectives, minimizing the make-span and total tardiness. This is a three-phase search algorithm, using heuristics and local search method. The first phase relocates the assigned set of the jobs to other machine. Secondly, the sequence of jobs changed for a machine and in the third phase, simultaneously, the set of jobs on machine and sequence of jobs is changed. Disposal hull approach and triangle method, were applied to ensure quality solutions. Costa et al. (2014) considered hybrid flow shop parallel batching (HFSPB), which involved stages with parallel and proposed a mixed linear programming model with a GA, which uses crossover operator for scheduling the jobs to minimize the make-span. Bassedik et al. (2015) investigated a novel hybrid immune system based on genetic algorithm with an aim to minimize the make-span for permuted flow shop scheduling environment and proposed two aspects, first being the hybrid of genetic algorithm and immune system, introduced a vaccination biological aspect of immunity, and second, based on immune network theory.

1.2 HEURISTICS

Heuristics refers to as an approach to problem solving, or discovery that employs a practical method not guaranteed to be optimal, but sufficient to obtain results. Polya (1945) gave some heuristics like working backward if unable to find a result, considering an example for an abstract problem, or picturing a problem rather than mathematically or theoretically focusing on it. Simon (1955) defined a term satisficing, which implies to the situation where further optimization of solutions is possible. Kahneman et al. (1982) studied the judgments made under uncertain situations. Heuristic methods can rapidly process the problem giving a satisfactory solution. Judea (1983) described heuristics as the strategies, influenced by experiences with resembling problems, using readily approachable, though loosely applicable information to control solving in machines, man and abstract issues. So heuristics simply

explains how people make decisions, come to judgments, and solve problems typically when facing complex problems or incomplete information.

1.2.1 Different Types of Heuristics to Solve Scheduling Problems

The researchers have found many methods to solve scheduling problems, applicable to the different manufacturing industries. Figure 1.2 depicts the various methods of solving scheduling problems.

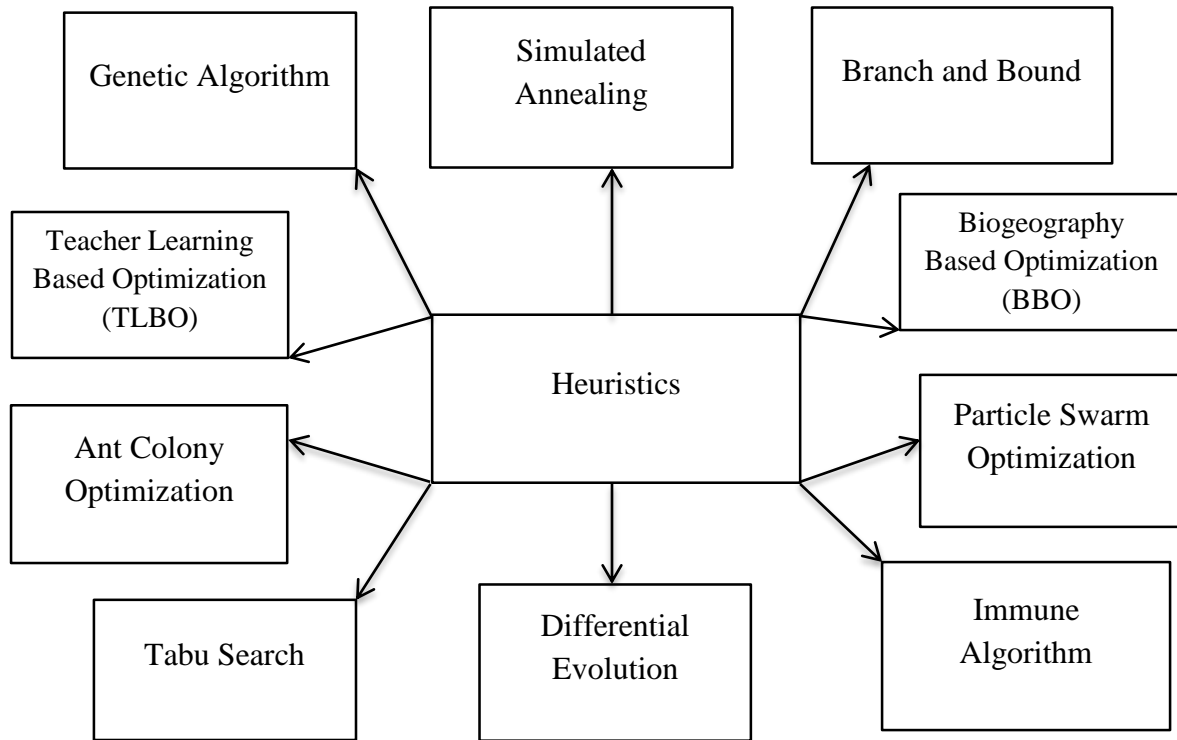


Figure 1.2: Depictions of Various Methods to Solve Scheduling Problems

1.2.2 Genetic Algorithms

Genetic Algorithm (GA) inspired and originated from the natural selection, is a meta-heuristic approach to produce high quality results in order to achieve optimization by using crossover, inversion, mutation and selection operators. GA is an approach to produce offsprings from the parent population called chromosomes, which consists of a gene. The selection operator, selects the fitter chromosomes to reproduce, crossover, interchange the two chromosomes, mutation, randomly vary the gene values in between chromosomes thus maintaining the diversity in new population and inversion, rearranges the genes in their respective arrayed order.

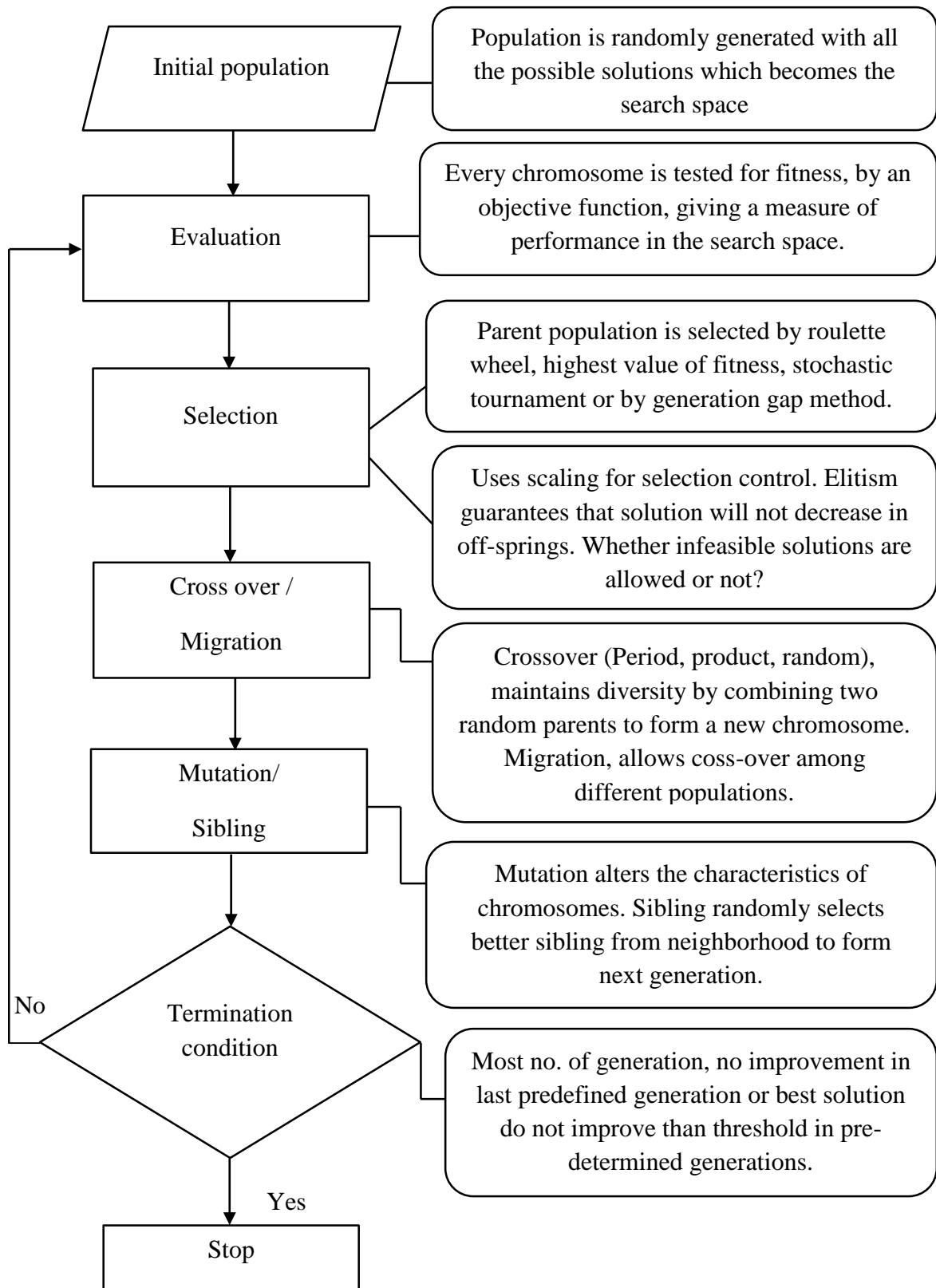


Figure 1.3: Flow Process Chart Depicting Steps in GA for Optimization

Chastang and Gremy (1978) verified compatibility of father's phenotype keeping the assumption of assured motherhood w.r.t the genotypes of all the children. Pettit and Pettit (1987) experimented in different noise environments (low, medium and high) to construct an adaptive approach to perform rapid classification of messages from sensor (possibly very noisy), w.r.t the objects present in the environment, so that continuous acquisition of knowledge takes place. Stark (1990) proposed a GA for a set of counts in classes, to compute standard error of estimator based on formula of R.A. Fisher, which involves their partial derivatives with respect to the frequencies, thus eliminating the need to compute them analytically. Lim (1997) proposed an efficient algorithm for material handling in a single hoist scheduling to determine cyclic schedules for printed-circuit-board (PCB) electroplating, maximize the line throughput rate. Alexouda and Paparrizos (2001) presented a GA, which consists of GA1 (uses random population to initialize) and GA2 (includes solution of beam search method in the first population) to solve product line design problem taking seller's marginal return as a criterion. Zhao et al. (2009) considered problem of travelling salesman for pickup and delivery of one commodity and proposed a GA based crossover operator utilizing both global and local information designing a new tour to generate initial population giving a faster convergence. Lu and Huang (2015) investigated cutting stock problem with two-dimensions in liquid crystal display industry and proposed an efficient and effective GA involving mixed integer programming model and corner space algorithm to reduce the production costs and improving the customer satisfaction. Morini and Pellegrino (2016) employed a genetic algorithm to explore a huge combinatorial space of tax structures with an aim to establish a best real world income tax structure, blocking the taxpayers for being worse off with present tax structure and maximizing redistribution of tax. Figure 1.3 shows the various steps involved in genetic algorithm in a hierarchy along with the functions/work each step performs to determine the result.

1.2.3 Simulated Annealing

Simulated Annealing (SA) published by Metropolis et al. (1953) is a probabilistic approach and is a bi-product of Monte Carlo method to determine states in thermodynamic system. It approximates the global optimization in a huge discrete search space. The name derived from metallurgy, which includes heating and cooling in a controlled manner to reduce the defects in a material and increasing its crystal size. The characteristic property is to accept the worse

solutions while exploring the search space and at the same time, probability for considering the worse solutions decreases with the cooling speed getting slower. Khachaturyan et al. (1979) formulated a simulated annealing approach to minimize function, having large number of variables in artificial multi-atomic system, to statistical equilibration of the system. Molitor (1985) proposed heuristic method using SA for minimizing the two laterals, being similar to crystallization process, to accept the arrangements, that lowers the function cost and regulate the uphill steps. Dolan et al. (1990) studied the efficiency of SA algorithm and implemented in heat exchanger network design, with the aims of calculation of the cost change between different states randomly generated, scheduling the annealing temperature and at-last determining the computational time. Berthiau and Siarry (1993) formulated an Enhanced SA Algorithm to address continuous variable problems of electronic component to reduce the computational, minimizing the open circuit simulator parameters and determine synaptic coefficients of analogue neural network. Chardaire (1993) proposed a heuristic, which is an extension of SA algorithm for 0-1 optimization problems, which approximates the variable value at a given temperature, fixing the variable as temperature decreases, thus reducing the problem size, so that better solutions are determined in less time. Raaymakers and Hoogeveen (2000) proposed a SA algorithm for multipurpose batch process scheduling environment with no-wait restrictions, which needs adaptation from neighborhood structure due to presence of overlapping operations to obtain near-optimal solutions to determine make-span. Varadharajan and Rajendran (2005) presented a multi-objective simulated annealing algorithm for permutation flow-shop scheduling to minimize the flow-time, make-span of jobs and, obtained three improvement schemes with two initial sequences containing minimum value of objective function, taking temperature and epoch length as parameters. Muppani and Adil (2008) developed a SA algorithm to resolve an integer programming model considering all aspects of combination of products, cost of storage space and cost of order picking for the class formation with cubic per order index restrictions. Czapinski (2010) presented parallel SA with genetic enhancement based on clustering algorithm introducing the new mechanism for dynamic parameters, which gives new solutions, along with those given by other meta-heuristics, in permuted flow shop environment to minimize the total flow time. Jayaswal and Agarwal (2014) addressed research on source dependent U-shaped assembly line balancing and proposed meta-heuristic based on SA, to determine

the optimal solution for small and medium problems eliminating large inventories, line inflexibility and monotony. Matai (2015) proposed a modified SA algorithm for multi-objective facility problem for any number of objectives, eliminating the dependency on decision maker to generate the weights to each objective. Shaabani and Kamalabadi (2016) presented mathematical model called population based simulated annealing for multi-period multi routed inventory routing problem which includes perishable products with fix life time, utilizing the shipping strategies with an aim of minimizing the total cost without any stock out condition.

1.2.4 Branch and Bound Method

Branch and Bound (B&B) proposed by A.H. Land and A.G. Doig in 1960's, and is an algorithm to find the solutions for combinatorial problems, general real valued and discrete problems. The aim of B&B is to determine a maximum or minimum value objective function. Three operations included are branching, i.e. producing subsets for solution, bounding, computing a lower bound against any candidate and solution, determining whether obtained result is feasible or not. It is a state space search, in which the solution formed as a rooted tree, then traversing the branches of the tree, which are subsets of a solution, are crosschecked by the upper and the lower bounds, hence giving an optimal solution. Henley and Williams (1973) described B&B as a powerful and feasible method against the time consumption and storage memory, a computer needs with the increasing number of variables to solve combinatorial problems. B&B includes search trees, whose each node delivers a possible solution corresponding to the cost bounds. When a new solution with respect to cost could not be less than the lower bounds, search stops. Bilde and Krarup (1977) developed a theorem, and presented algorithms to determine lower bounds using B&B for solving un-capacitated plant issues including set partitioning and set covering. Also, gave a computational report showing their high standard of performance and simplicity. Henry and Penny (1982) presented two efficient B&B algorithms and implemented for practical problems for extracting minimal and near-minimal phylogenetic trees from protein sequence data. Christofides et al. (1987) described a B&B algorithm, for project scheduling environment where conflicts arises due to the scheduling of sets of activities constrained by the availability of the resources. There were four lower bounds determined, first based on longest path computation, second, based on linear programming relaxation, third, on

Lagrangian relaxation and fourth based on disjunctive arcs to plot the problem on the graph. Quesada and Grossmann (1992) proposed an Linear programming based B&B algorithm to improve the efficiency of MINLP problems by defining the master problem dynamically during the tree search reducing the number of nodes and thus, predicting the lower bounds by solving linear programming sub-problems, until appropriate solution is found. Yamada et al. (1997) developed B& B algorithm with three lower bounds for mini- max spanning problem in forests, to reduce the costs of constituent trees by finding a spanning forest of an undirected graph. Sun (2002) formulated two B&B algorithms, which uses primal network simplex method to determine relaxations in network and Driebeek penalties to strengthen lower bound to select variables, to solve a logical and distribution transportation problem with side constraints, is modeled as mixed integer programming model in 0-1 coding. Mu et al. (2007) presented simple, efficient and fast B&B algorithm for binary quadratic programming problem with pre-treatment, which initially decreases the size of programming and then, gave a new pruning method, bounding method and procedure for initial solution. Kellegoz and Toklu (2012) proposed B&B algorithm based on efficient feasibility and dominance criteria, and heuristic for enumeration process, to utilize large area of line-established, investment (of equipment's and tools) and work-in-process in assembly lines of long product flow line with multi-manned parallel work stations. Ozturk et al. (2014) presented high quality B&B algorithm to solve problem of parallel batching in hospital sterilization services, where jobs have different release dates, sizes and same processing times with an objective to minimize the make-span. Abdelsadek et al. (2015) presented a B&B using lower bound supported by greedy search and genetic algorithm, for undirected graph G to determine maximum un-weighed vertex-triangles and, also, proposed an upper bound based on surrogate relaxation. Freire et al. (2016) re-formulated the capture demand problem with random utilities into Mixed-Integer Linear Programming model and proposed B&B algorithm using greedy approach to determine relaxation and, applied it in park-and-ride facility location.

1.2.5 Ant Colony Optimization

Ant Colony optimization (ACO) proposed by Dorigo (1992) is a probabilistic method to solve combinatorial problems by determining the paths through graphs. Dorigo and Gambardella (1997) described ant colony system, based on the natural ant behavior, in which

upon return after finding food, to their colony an ant leaves pheromone trail and other ants follow that reinforcing if they eventually find food. The evaporation of pheromone is critical, as it avoids the convergence of local optimal solution, and depends on the path lengths, if longer, pheromone evaporates for more time and if short, more ants can travel frequently, thus increasing the density of pheromone. The aim is to mimic the behavior of the simulated ants giving a positive feedback and represents problem to solve, on graph. Sendova-Franks and Franks (1993) reviewed the division of labor with monomorphic worker environment, also, presented a data for ant colony of small size in a seasonal environment, which resulted in workers ability to reproduce with existing variety of task. Dorigo and Gambardella (1997) demonstrated the capability of the quality of solutions for both symmetric and asymmetric approaches for travelling salesman problem by computer simulations and found that Ant Colony Optimization technique generates feasible solutions in shorter periods. Song et al. (1999) presented a novel ant colony search algorithm for combined heat and power economic dispatch lines (with multi-objectives), and listed the three characteristics, positive feedback, constructive greedy heuristic and distributed computation. T'kindt et al. (2002) proposed ACO approach, by using simulated annealing feature and local search algorithm for two-machine flow-shop scheduling environment with aim of minimizing make-span and total completion time. Shyu (2004) listed the applications, and proposed an ACO algorithm with several features for flow-shop environment where no in-between storage is available between two workstations and each operation needs a different setup, with an objective of minimizing total completion time. Hani et al. (2007) presented a hybrid ant colonization approach combined with guided local approach in a quadratic assignment problem in train maintenance facility and demonstrated improvement in the new layout that the actual layout. Gajpal and Abad (2009) proposed a Multi-Ant Colony System (MACS) to design the routes for vehicles in a Capacitated Vehicle Routing Problem (CVRP), constrained by the number of vehicles and deliveries to line-haul customers being the priority, in order to minimize the total distance travelled by the vehicle. Liao et.al (2014) proposed a Unified Ant Colony Optimization Algorithm (UACOR) for continuous optimization, which is capable of producing new ACO algorithms by using automatic configuration techniques, and presented two new algorithms, UACOR-s and UACOR-c. Bera and Mukherjee (2016) presented a combined ACO algorithm with a desirability function for

serial multi-stage manufacturing system in multi-response optimization problem, with an aim of optimal product quality constrained by setting condition at each stage having multi-correlated responses at each stage by building empirical response function.

1.2.6 Particle Swarm Optimization

Particle Swarm Optimization (PSO) proposed by Kennedy and Eberhart (1995) is a meta-heuristic approach, to optimize the problem by taking candidate solution as an input and moving them around the search space by its velocity and position and is influenced by their own and entire swarms (population) best-known position, and guided toward best positions, thus upgrading better positions. PSO can search large spaces with fewer assumptions and do not require differentiable function. Abido (2002) presented efficient and reliable PSO for solving the optimal power flow with aim of voltage profile improvement, fuel cost minimization and voltage stability enhancement, by incorporating PSO, derivative-free technique for optimization controlling the optimal setting variables. Yin and Wang (2006) presented an effective and efficient PSO for solving a nonlinear resource allocation problem, by instructing the problem using proposing adaptive resource bounds for optimally allocating the limited number of resources to the various activities. Zhang et al. (2008) proposed an improved PSO algorithm to resolve large-scale flow shop scheduling problem, which aim at minimizing the make-span by combining genetic operators and PSO algorithm, in which mutation operator searches neighborhood and is all different constrained. Yalaoui et al. (2009) presented a hybrid PSO algorithm and GA to solve facility layout problem, for group technology environment, which simultaneously include machines and products in a manufacturing cell and regarded as quadratic assignment problem. Unler and Murat (2010) investigated feature subset selection problem using a regression model and employed a discrete PSO algorithm, which dynamically accounts dependency and relevance of features included in the problem subset, and find it to be competitive with scatter search and Tabu Search algorithm. Wille et al. (2011) proposed an efficient discrete PSO algorithm for discrete capacity assignment problem in a realistic general-topology network, which is constrained by end-to-end quality of service, for obtaining optimal solutions with small computational effort. Altinoz and Yilmaz (2012) formulated a PSO algorithm with parameter dependency walls in synthesis problem of microstrip-like interconnected lines with an aim of finding rapid solutions for optimization problems where dependencies exists in-between

input variables. Prescilla and Selvakumar (2013) applied a Binary PSO algorithm for the real-time task assignment, which includes several independent periodic tasks, in a heterogeneous multiprocessor with an aim to schedule maximum tasks and consuming minimum energy without surpassing the utilization bound. Lin et al. (2015) proposed a novel Multi Objective PSO (MOPSO) for multi-objective optimization problems, which aggregates to a single problem and then assigning each particle to every aggregation problem, also, designing two search strategies for updating velocity of each particle to increase convergence speed and maintain the population diversity. Kumar et al. (2016) developed a Self Learning PSO (SLPSO) algorithm to solve Multi-objective Multi-Vehicle Production and Pollution Routing Problems with a Time Window (MMPPRP-TW) which focuses on optimizing the routes of vehicles in supply chain planning system to minimize emissions and total operational cost.

1.2.7 Tabu Search

Tabu Search (TS) invented by Glover (1986) is a local search method used for optimization. It approaches the neighborhood solution, which means the similar solutions except with minor details to generate a new improved solution. Ability to accept the worsening moves on sub-optimal and plateaus, where solutions are likely to fit equally and, introducing the prohibitions to stop the search from coming back to yester visited solutions, enhances its performance. Glover (1989) described the memory structures, short term, intermediate and long term to store the visited solutions. Hertz (1991) adapted TS for large-scale time tabling problems to minimize the conflicts due to courses taking place parallel to each other, which include common teachers or students requiring the same classroom, which are constrained by groups of students, geographical and precedence requirements. Icmeli and Erenguc (1994) proposed a TS algorithm for Resource Constrained Project Scheduling Problem with Discounted Cash Flows (RCPSDC) with an objective to maximize net present value with precedence and resource constraints, producing optimal solutions with reasonable computational effort. Armentano and Ronconi (1999) proposed a heuristic based on TS for permutation flow-shop scheduling problem with an aim to minimize total tardiness, including the evaluated strategies of intensification, diversification and neighborhood restriction and explored the solution space to obtain better results in reasonable period. Chao (2002) constructed and tested TS method using the deviation concept found in annealing to solve

truck and trailer routing problem considering the three routes, a pure truck route by truck alone, a pure vehicle route without any sub-tours and a complete vehicle route with sub-tours, with an objective to minimize the cost incurred and total travel distance. Grabowski and Wodecki (2004) proposed a fast and accurate local search procedure based on TS method for permutation large-size flow shop scheduling problem with an aim of minimizing the make-span, by using lower bounds on make-span instead, computing explicitly and introduce perturbations for exploring the search space for better solutions. Mika et al. (2008) proposed a local search meta-heuristic based on TS for a multimode resource constrained project-scheduling problem including the schedule dependent set-up times with renewable resources and pre-emptable activities with an aim to minimize the total duration of the project. Lamghari and Dimitrakopoulos (2012) presented a meta-heuristic approach based on TS for open-pit mine production scheduling problem constrained by metal uncertainty, using two diversification strategies, exploiting the long-term memory of search history and neighborhood search strategy, producing solutions in short computational times. Costa et al. (2015) presented a Parallel Tabu Search (PTS) algorithm along with neighborhood generation mechanism in a serial production system for a buffer allocation problem, with an aim to minimize the total buffer capacity and developed a multi-factor analysis to analyze the influencing factors, from the results obtained. Sukkerd and Wuttiornpun (2016) proposed a hybrid algorithm of TS and GA, which includes various steps, generating the production schedule, generate initial sequence of orders, iteratively improving the sequences and determining the start time of operations, in finite capacity material requirement planning system in a flexible flow shop environment with assembly operations, to minimize the computational time.

1.2.8 Differential Evolution

Differential Evolution (DE) is a meta-heuristic, and an optimization method, which improves the solution with respect to the desired quality, required. The characteristic feature is the non-requirement of gradient of the problem i.e. a problem does not need to be differentiable, thus allowing its applications in noisy, vary with time and are not continuous. DE generates new solutions by collaboration of existing ones and stores the candidate solution depending upon the fitness or best score, when compared among each other. Bahu and Shastry (1999) combined DE with orthogonal collocation proposing a non-sequential technique for

estimating the effective heat transfer parameters in trickle bed reactors, assuring the convergence from any initial point with reduced function evaluations, in a less computational time as compared to radial temperature profile. Kyprianou et al. (2001) used DE method in memory dependent vibrations system to carry out the direct search without any derivative estimation in a continuous parameter space, to simulate the noisy data, noise-free data and experimental data of a nuclear power plant. Cruz et al. (2003) investigated DE algorithm for solving multimodal optimal control problems and claimed it to be efficient, robust and effective and at the same time eliminating the drawback of long computational times, dependency on gradient information in other evolutionary techniques. Nearchou (2005) described the application of DE algorithm in assembly line balancing problem, with an aim to minimize the number of workstations needed to manufacture a product constrained by fixed cycle time. Al-Anzi and Allahverdi (2007) presented a self-adaptive DE heuristic for two-stage assembly flow shop scheduling problem including the set-up times in processing times with an aim to minimize the maximum lateness and compared the results with and outperforming particle swarm optimization by reducing computational time by one-third. Damak et al. (2009) proposed a DE algorithm for resource constrained project scheduling problem including multiple execution modes with an objective to minimize the make-span and computing the results in less time for each activity. Dong and Wang (2012) developed a novel Hybrid Permutation Based DE (HPDE) for Zero Wait Scheduling Problems (ZWSP) specified as, asymmetric travelling salesman problem, resulting in high quality solutions with reduced computational time requiring less user-defined parameters, making it applicable to real life large scale ZWSPs involving set up times. Ying Liu et al. (2014) proposed a hybrid DE by combining DE with Individual Improving Scheme (IIS), which diversifies the population with enhanced solutions, and greedy based local search which guides algorithm to escape local minimum, for permutation flow shop scheduling problem. Zhou et al.(2016) formulated an effective DE based hybrid algorithm for scheduling uniform parallel batch processing machines having different capacities with arbitrary job sizes, by first, representing the population as discrete jobs and applying crossover, mutation operators, secondly, forming the batches and scheduling them on uniform parallel machines, with an objective to minimize the make-span.

1.2.9 Immune Algorithm

Artificial Immune System (AIS) authored by Farmer (1986) is a technique intended to function and mechanize as immune system do, to solve the computational problems from engineering, mathematics and information technology. AIS is an adaptive system and bi-product of natural computing and biological inspired computing considering immunology for its principles, models and working. Huang (1999) proposed IA for thermal generation scheduling problems, in which constraints and objective functions were made antigens, and antibodies were determined by calculating the affinity which is the deemed solution to the problem. Wen and Song (2004) proposed an immune evolutionary algorithm (IEA), which consist of self-adaptive mutation operator to decide step size of antibody and affinity calculation, to maintain diversity, to evaluate the sphericity error by making objective function as antigen and fitting the antibodies with antigen to find the solution. Zandieh et al. (2006) described an overview and basic notions, and presented an IA for hybrid flow shop scheduling problems with sequence dependent set-up times, to obtain a manufacturing schedule within a reasonable time and then compared the results to random key genetic algorithm, which is out-performed. Hsieh et al. (2009) proposed an IA for flow shop scheduling problem with buffers with an aim to minimize the make-span and the solutions proved to be superior as compared to hybrid genetic algorithm with less relative errors, also, showed the impact on make-span with and without buffers. Wang et al. (2009) proposed an immune-genetic algorithm for planning of the new products, which is a type of semi-infinite programming model having infinite constraints, by first generating random antigens along with training antibodies, then applying immune system to recognize self and non-self antigens by antibodies, and at-last, repairing the infeasible chromosomes. Massim et al. (2010) implemented a combined artificial immune system optimization algorithm with decomposition method with an aim to allocate buffers in transfer lines in order to maximize the line throughput, optimize work in process inventory and maximizing the economic profit. Lin (2013) presented a novel immune multi-objective optimization algorithm, based on micro-population involving a novel adaptive mutation operator (applied according to the fitness value) and fine grained selection operator for preserving the population diversity, thus enhances the ability to converge. Azadeh et al. (2014) proposed combined algorithm of artificial immune system with genetic algorithm and particle swarm optimization, for

forecasting yearly electrical energy consumption by utilizing the fittest random variables as an input to reduce the relative error and used mean absolute percentage error to evaluate the results to select forecasting model. Souza et al. (2016) used artificial immune algorithm for combinatorial optimization to reconfigure the electrical distribution systems with a varying demand, with an objective to determine the radial topology and to minimize the cost of energy losses for given operation time.

1.2.10 Teacher Learning Based Optimization (TLBO)

TLBO is an optimization method, proposed by Rao et al. in 2011 which is based on the teacher and student learning process. It is a naturally inspired population method, where class of learners will represent the population. The best learner in the process is selected as a teacher, as only a teacher is considered with best knowledge and then increments the knowledge level of the students known as learners, so as to obtain the good marks. Here, the capability of a teacher to deliver and the quality of the class present also plays an important factor in order to increase the average of the class. There are two phases which constitutes the whole process namely, teacher's phase i.e. grabbing knowledge directly from the teacher and learner's phase, which motivates the grabbing knowledge between the learners. In the teacher phase, the teacher approaches to impart all of his knowledge among the class which is impractical in reality. This is because of the difference in the capability of delivering by teacher and that of understanding by the students. The learner phase on the other hand, inputs the knowledge from teaching phase and then further, increases it by interaction among the learners. Zou et al. (2013) proposed TLBO method for multi objective optimization problems by utilizing crowding distance computation mechanism with non-dominated sorting method. The highest crowding values are selected as the teacher and the centroid of the non-dominated solutions is the mean of the learners. Baykasoglu et al. (2014) investigated the performance of TLBO for optimization problems along with constrained and unconstrained linear programming problems for flow shop scheduling environment. The objective studied was to minimize the makespan constrained by precedence relations. Also, the comparative statistical analysis with the other methods has been provided to show the superiority of TLBO. Chen et al. (2015) presented an enhanced TLBO by adding local learning and self-learning methods. In self-learning, the individuals renew their position or exploit randomly to the new positions whereas local learning maintains the diversity of the population by

rearranging the set of iterations. Patel and Savsani (2016) proposed the teacher tutorial and self-learning in TLBO for multi-objective optimization problems of Sterling heat engine. The non-dominated solutions are assessed by grid based approach and the objectives undertaken were maximizing the thermal efficiency, output power and minimizing total pressure drop of the engine.

1.2.11 Biogeography Based Optimization (BBO)

Biogeography is inspired from the nature's geographic dispersion and proportioning of the biological organisms and was formulated by Dan Simon in 2008. BBO imbibe features of genetic algorithms and particle swarm optimization therefore can be utilized for the same problems these two. BBO is capable of laying down the mathematical models for migration of the species and their extinction along with the rise of new species. This is done in order to relocate the population of species to the neighboring islands. There are two phases namely, migration and mutation. Mutation phase maintains the diversity in the population. The term island refers to the habitat which has been isolated from the other habitats. For population to grow, and it is supposed to have high suitability index (HSI) which is dependent on the natural conditions such as rainfall, temperature, topography and vegetation whereas suitability variable index is independent of the conditions. High HSI will lead to emigration of various species to the nearby habitats by virtue of large species they host. Low HSI habitat experiences high immigration rate due to their sparse population and results in increase of the HSI. Bhattacharya and Chattopadhyay (2010) proposed BBO for solving the convex and non-convex economic load dispatch problem under taking constraints such as transmission losses, ramp rate limits, multi-fuel options and prohibited operating zones. The proposed method was applied to a 6-generator system limited by ramp rate, 40 generators with valve point loading and 10 generators with multiple fuels availability. Roy et al. (2012) presented multi-constrained optimal reactive power flow (ORPF) problem in power system with an objective of bus voltage deviation and real power loss. When the presented method was compared with other population based methods, the former founds better results. Wang and Duan (2014) proposed a hybrid biogeography based optimization heuristic for scheduling problems. Two strategies namely, chaos strategy and searching around the optimum strategy were embedded in the BBO mechanism in order to stabilize the value of global optimum. Santosa and Safitir (2015) developed BBO heuristic to solve the combinatorial problem of

single machine scheduling environment with an objective of minimizing the total weighted tardiness. When computed, BBO was able to solve 57 out of 75 instances while PSO solved only 19 out of 75 instances, hence showing the superiority of the method. Lin and Zhang (2016) investigated distributed assembly permutation flow-shop scheduling problem (DAPFSP) by using hybrid biogeography based optimization (HBBO) with an objective of optimizing the makespan for the manufacturing system. The migration phase was inserted with path relinking mechanism and mutation phase utilizes an insertion based method.

1.3 MULTI-OBJECTIVE FLOW SHOP SCHEDULING PROBLEMS (MFSP)

In the scheduling problems, there exists many objectives such as minimizing the make-span, tardiness, lateness, flow-time, achieving due dates; decreasing job disruptions, energy consumption etc. There are single objective problems, which only solve one objective, bi-objective problems, which aim to solve two objectives and multi-objective problems, which considers more than two objectives. MFSP problems are more complex and considered as NP-Hard (non-deterministic polynomial time) whose exact solutions do not exist. Loukil et al.(2007) did a case study of scheduling environment where first sub-parts are manufactured, which are followed by the assembly of final product and proposed a multi-objective simulated annealing approach with objectives of minimizing make-span, maximum tardiness, mean tardiness and completion time. The paper considered the overlap of processing periods of two successive operations of same job. Qian et al. (2009) proposed a hybrid algorithm based on differential evolution for permutation flow shop scheduling problem with multiple objectives and constrained by limited buffers between consecutive machines. First, largest order value (LOV) rule was employed to convert continuous values in DE to job permutations and then local search method is applied to explore the solutions and Pareto dominance was used to update the solutions. Yagmahan and Yenisey (2010) presented a multi-objective ant-colony system algorithm (MOACSA) for flow shop scheduling with objectives to minimize make-span and total flow time. Local search strategy, which explores the solution space was combined with ant colony algorithm and performance was compared with multi objective heuristics, showing MOACSA being more efficient. Azadeh et al. (2015) investigated flow shop manufacturing system having multi state machines with multiple objectives, such as, maximizing system reliability, minimizing make-span and

minimizing the purchasing cost by proposing a genetic algorithm to find near optimal solutions. Han et al. (2016) proposed a novel multi-objective optimization algorithm using GA to solve blocking lot-streaming flow shop scheduling problem, in which differences among parents and non-dominated solutions are used to design crossover operator and local search strategy was employed to explore the search space. Tang et al. (2016), proposed a novel particle swarm optimization approach to address flexible flow shop scheduling problem, to achieve objectives of minimizing the disruptions in job arrivals, energy consumption and minimizing the make-span (constrained by machine breakdown), hence meet the demand of sustainable manufacturing.

1.4 SEQUENCE DEPENDENT SETUP TIMES (SDST)

In scheduling, set-up time makes problem more complex and comes to play when production changeover is required between the different jobs, taking different amount of time to set-up on the machine before starting the operation. There are two type of structures; simple, in which set-up is independent of sequences and decisions for previous times, and complex, in which set-up time is dependent on both the factors. There exists three types of complex structures; first includes set-up carryover, hence allowing non-disruptive production run from last time to present without any additional set-up, second, contains a major set-up for similar jobs and third is dependent on the production sequence. Kim et al. (1996) proposed Tabu Search (TS) and Simulated Annealing (SA) algorithms for flow-shop environment in a printed circuit board industry, with lot streaming and sequence dependent setup times considered for each lot. The aim was to minimize the mean tardiness as there was time-lag between the machines because it was possible to start job on following machine before job is entirely completed on previous machine. Rios-Mercado and Brad (1998) presented a branch-and-cut (B&C) algorithm for flow-shop scheduling problem with sequence dependent set-up times with an aim to minimize the make-span by formulating two models, Model A, based on salesman travelling problem and Model B, which is less structured and uses few binary variables and constraints. B&C performed better than Branch and bound approach. Ruiz et al (2005) applied genetic algorithms to check their effectiveness in a permutation flow-shop scheduling problem with sequence dependent setup times with an aim to minimize make-span. Design of Experiments (DOE), was used for calibrating the parameters and operators

and evaluated the results with benchmark based on instances of Taillard. Ruiz and Stitzle (2008) proposed two Iterated Greedy (IG) algorithms for complex flow-shop problem with sequence dependent setup time on machines with objectives of minimizing total weighted tardiness and maximum completion time. One IG algorithm is based on adaption of IG principle and second, based on local search approach. Kirlik and Oguz (2012) presented effective, efficient and robust General Variable Neighborhood Search (GVNS) heuristic to solve problem of sequence dependent setups with an aim to minimize total weighted tardiness. For single machine scheduling with varying size from small to large, 35 instances were optimized and improved results of 16 instances, out of total 64 instances considered. Ciavotta et al. (2013) proposed a novel method named as Restarted Iterated Pareto Greedy (RIPG) approach for sequence dependent setups in permutation flow shop scheduling problem, both with single objectives and multi-objective flow-shop problem. Pareto approach and Greedy method was combined and when tested, outperformed conventional approaches. Jeong and Kim (2014) presented a Branch and bound based algorithm and heuristic algorithm for two-machine re-entrant flow-shop scheduling problem with objective of minimizing total tardiness. Each job was twice processed, and considered sequence dependent setup times on the second machine for developing dominance properties and a lower bound used in branch and bound based algorithm.

1.5 BACKLOGGING IN SCHEDULING

In the manufacturing industry, backlog is the uncompleted, unprocessed work for a specified time or jobs in the process of completion. It implies to the workload, which is beyond the capacity of the production system. The factor on which it depends is waiting time more the waiting time lower is the backlogging rate. Partial backlogging is a situation where the demand of a product met from other sources where as in full backlogging, demand remains unfulfilled until the next order. Yan et al. (2010) tested a hypothesis for a case of internet retailers with objective of managing the backlog variation in the order fulfillment to meet supply, capacity and efficiency of the system by studying the link between backlog variation and order fulfillment. Wu et al. (2011) proposed mixed integer linear model for capacitated multi-level lot sizing problem, constrained by backlogging with an aim to provide lower bound on optimal solutions. Brito and de Almeida (2012) presented multi-attribute utility

model for newsvendor model with an objective of making explicit decisions for order quantities, impacted by backlogging, in order to maximize the profit. Ouyang and Chang (2013) presented a mathematical model for reworking of imperfect items and trade cost due to imperfect production processes and backlogging with an objective to reduce cost of production inventory by reworking them, which incurs less holding cost. Babaei et al. (2014), developed a genetic algorithm for capacitated scheduling and lot sizing problem with sequence dependent setups, backlogging and setup carryover to study the complexity and determine the near-optimal solutions in reasonable computational time, and tested its accuracy by developing a lower bound.

CHAPTER 2: LITERATURE REVIEW

2.1 INTRODUCTION

This chapter reviews the contributions made towards multi-objective flow shop scheduling problems, sequence dependent set-up times problems and backlogging problems in the past few years. As flow-shop environment is worldwide practice in the manufacturing industries, hence important to improve the productivity to achieve profit-worthy status in the economy. The significance of reviewing lies in the diversity of flow shop problem, its parameters which on optimizing/minimizing even one of them would produce significant results in the efficiency and effectiveness of the production system. Each review describes the method/approach/heuristics used to solve the problems with their specified objectives, how the method works in order to produce results and the software used to code the algorithms along with the comparisons made with the respective contemporary methods. The introduction chapter showed the spread of flow shop problems in the different fields of science and many heuristics to compute results, which here has been narrow to few only. One most used heuristic is genetic algorithm and objective to be achieved is to minimize the makes-pan in the below contributions.

2.2 RESEARCH MODELS

Azzi et al. (2012) presented a heuristic procedure to solve hybrid flow shop scheduling problem in a flexible multistage batch production system with an aim to minimize make-span. The other aim was to increase production capacity utilization (affected due to decrease in marketing time and increased number of models) by using a batch splitting or aggregation strategy and introduced workload leveling function (which determines the cost by reducing the make-span by rearranging the machines or by splitting into batches) as each stage comprises of certain number of machines. This condition is acceptable when benefits considered be function of setting up new machines and reducing process time and when same job is required at the different stages of production batches, set-up times be reduced. The system includes two or more production stages and a job processed on one, none or every machine groped in the batches with buffers of work in progress among different stages. When compared with short processing time method, reduced make-span by 65 % and increased capacity to 60 %. The proposed heuristic has a potential to reduce make-span with

inter-operational buffer limits in multi-stage multi-product batch production system with first-in-first-out flow.

$$\text{Liv}(m_1, m_2, \dots, m_h, m_z, j_z) = \text{PT}(m_z) - \text{PT}(m_1) - \text{TS}(j_z)/H + 1 \quad (2.1)$$

$$\text{PT}(m_{i_{\text{new}}}) = \text{PT}(m_i) + \text{TS}(j_z) - \text{Liv}(m_i, m_j, j_z) \quad (2.2)$$

$$\text{PT}(m_z)_{\text{new}} = \text{PT}(m_z) - \text{Liv}(m_i, m_z, j_z) \cdot H \quad (2.3)$$

$$\text{PT}(m_z) - \text{PT}(m_1) > K \cdot \text{TS}(j) \quad (2.4)$$

Javandi et al. (2012) proposed a mathematical model and immune algorithm for hybrid flow shop scheduling problem with sequence dependent set-up times, time lags and possibility of jobs to skip stage, with an aim to minimize make-span in solving small size problems in a reasonable computational time. The algorithm uses mutation to search near optimal solution and redeploys the best solution in the subsequent population, thus striking out the solution with better convergence. Immune algorithm outperformed the presented mathematical model in producing optimal solutions in less computational time. The proposed algorithm was coded in MATLAB 7.5 and solved in LINGO 8.0 in Pentium IV 2-GHz processor with 1GB RAM.

Singh and Mahapatra (2012) presented Particle Swarm Optimization (PSO) algorithm to solve flexible flow-shop scheduling problem, which combines both flow shop and parallel machines in order to minimize the make-span so that multiprocessor system be executed with minimum length. The algorithm uses mutation operator to avoid premature convergence and trapping of solution in the local minima. Chaotic numbers generated with help of logistic mapping to converge the solution rapidly towards near-optimal solution, thus reducing the computational time. When compared, the average percentage deviation in PSO was 2.961 to that of 3.559 in genetic algorithm (GA), hence making PSO to be more effective in producing quality solutions in less computational time and with less parameters. The algorithm was structured in C++ on Pentium 4 3GHz processor with 1 GB RAM.

Elyasi and Salmasi (2013) proposed a dynamic method for flow-shop scheduling problem with an aim to reduce number of tardy jobs constrained by stochastic due dates and included processing times. The due dates followed normal distribution with pre-known mean and variance. The proposed method decomposes m-machine flow shop problem into sub-problems of m-stochastic single machine scheduling, each of which regarded as individual

mathematical problem. The presented method when compared to shortest processing time (SPT) proved 23.9% better on scheduling average industry-size problem and observed that more the environment is dynamic better were the results produced. The computational time of both the compared methods was same. This method has a potential to solve large size problems with random processing times or weighted jobs. C# language was used to code the algorithm along with LINGO 11.0 to solve the individual problems on 2 GHz Intel core processor with 3GB RAM.

Mousavi et al. (2013) addressed hybrid flow shop scheduling problem in just in time environment, to minimize total tardiness and make-span. Heuristics and local search method combined to develop an efficient bi-objective local search algorithm (BOLS) with three phases. First phase moves assigned set of jobs to other machines, second phase changes the order of machines and third phase simultaneously changes job set of machine and order of jobs. The quality of solutions verified by testing 30 instances with a maximum of 50 jobs using triangle method and hull approach proved to be better than multi-objective simulated annealing (MOSA) and bi-objective heuristic (BOH) approach. The presented method has a potential to find optimum solutions including total completion time, maximum lateness and to find Pareto frontier using other evolutionary algorithms. The algorithms were coded in MATLAB 7 with CPU 800 processor having 512 RAM.

Toledo et al. (2013) proposed a novel hybrid multi population genetic algorithm (HMPGA), which combines mathematical programming technique and meta-heuristic for multi population, using fix and optimize (FO) heuristic to solve capacitated lot sizing problem with backlogging. HMPGA produced three populations, which structured individuals in a tree. The genetic operators, crossover and mutation, explore the solution space of variables and FO intensifies its exploration in the neighborhood of better individuals in mix integer problem. The FO heuristic improve the following fittest individuals using two rolling horizon windows for fixing and optimizing variables indexed by periods and families, and acts as memetic component of hybrid method. When results compared with Akartunali and Miller heuristic (AMH) solutions, HMPGA outperformed it 75 out of 120 times and performed better in the sets of higher resource utilization in a reasonable computational time. The heuristics were coded in GAMS and computed in CPLEX 12.2 on Intel core 2-duo 2.66 GHz processor with 2GB RAM.

Vanchipura and Sridharan (2013) formulated two constructive heuristics named as Setup Ranking Algorithm (SRA) and Fictitious job setup ranking algorithm (FJSRA) in flow shop scheduling environment with sequence dependent set-up times, with an aim to minimize make-span. SRA has the ability to produce sequences from the set-up time of the jobs and FJSRA constitutes of jobs with minimum set-up times. Taillard benchmark problems at eight different levels were tested by proposed approaches with their graphical analysis, statistical analysis and relative performance index, and FJSRA proved to be better for large problems and small problems with higher set-up time. Drawback of SRA was not giving importance to processing time. The problem was coded in MATLAB using core2duo 2GHz processor with 2GB RAM.

Babaei et al. (2014) formulated a genetic algorithm (GA) using crossover and mutation operators, to address backlogging, setup carry over and sequence dependent setups for lot sizing and scheduling problems to find optimal solutions constrained by minimum computational times. The problem is a mix integer linear problem. GA combined with a procedure to obtain lower bound against optimal solutions garnered better results and the same lower bound use to evaluate the proposed algorithm. The future scope includes use of other meta-heuristics like particle swarm optimization, simulated annealing and imperialist competitive approach to solve this model. Some of parameters that can be involved in this problem are maintenance activities, machine breakdown, infinite buffers between levels and stochastic process time. Also, analysis of variance was performed and results when compared appealed for the effectiveness and efficiency of the algorithm. MATLAB was use to encode the algorithm and LINGO 8.0 in Intel core duo 2.94 GHz processor was used to frame the problem.

Chutima and Narumitwong (2014) presented a Pareto biogeography- based optimization (BBO) approach for multi-objective sequencing problem for two-sided assembly line (SAL) considering a learning effect with objectives of minimizing total sequence dependent setup time, minimizing total utility work and minimizing the variance of production rate. Adaptive BBO (A-BBO) was embedded into BBO for obtaining high quality non-dominated solutions in such a way that both A-BBO and BBO work together through monitoring feedback status of evolving solution and triggering adaptive commands to control parameters of algorithm. A-BBO explores and exploits the search space, converge the solution and avoids premature

convergence. The algorithm was encoded in MATLAB using Intel Core i7 2.20 GHz processor with 4 GB RAM.

Costa et al. (2014) proposed a mixed integer linear programming model and smart decoding-based genetic algorithm (SGA), to manage hybrid flow-shop (HFSP) environment, with parallel batching, limited machine capacity and eligibility machine restrictions. The objective was to minimize the make span. SGA uses permutation encoding scheme that manages batching of identical jobs, job sequencing and machine eligibilities. The crossover operator known as enclosed order crossover (EOX) keeps the set of identical jobs included in a solution, unchanged. The paper employed full factorial experiment to calibrate the parameters and selects the best combination to be used. Also, analysis of variance (ANOVA) demonstrated the effectiveness of SGA under both computational time and quality of solutions. This method contains potential to solve HFSP with stage skipping or sequence dependent setup times. The problem was framed using Design Expert 7.0.0 version.

Gerstl and Mosheiov (2014) developed dynamic programming algorithms for the scheduling problem of unit time jobs and batch production in a two-stage flexible flow shop environment, with aim of minimizing the makespan and total flow time. The study assumed, batch availability i.e. all the jobs in the batch be completed, batch consistency, which allocates same batches in both stages and non-anticipatory set-up times, which enforced processing on second stage only after all jobs of the batch are processed on first stage. Also, the sequence dependent set-up times are included in the study. Both, the stages of the flow shop environment, involves different number of parallel identical machines and no restriction exists, on the number of batches to process on each machine. Although the proposed approach are polynomial and guarantee optimal schedules, but is inefficient for large number of machines.

Huang et al. (2014) developed an effective and robust Farness Particle Swarm Optimization (FPSO) algorithm for re-entrant flow-shop scheduling problem, with objectives of minimizing tardiness and total weighted earliness, by taking into consideration wafer testing process with due window problem. Results when compared with those of Particle swarm optimization method and ant colony optimization, outperformed both in terms of average improvement of effectiveness by 33.47 and 42.99 %, respectively and in robustness by

44.44% and 39.36%, respectively for small-scale problems. For large-scale problems improvement in robustness concluded were 36.74 and 55.11% respectively. This shows the capacity of FPSO in solving scheduling problems. LINGO 13 used to determine mathematical results and heuristics in C++ using Intel core i5 2.81 Hz processor with 3.46 GB RAM.

$$\text{Objective function} = \min w_1 \sum_{j=1}^n E_j + w_2 \sum_{j=1}^n T_j \quad (2.5)$$

$$E_j = \max\{d_j^L - C_j, 0\}, j = 1, 2, 3, \dots, n \quad (2.6)$$

$$T_j = \max\{C_j - d_j^U, 0\}, j = 1, 2, 3, \dots, n \quad (2.7)$$

$$C_j = \max_{i \in \{1, 2\}} \{C_{ij}\}, j = 1, 2, 3, \dots, n \quad (2.8)$$

$$\sum_{k=1}^{m_i} x_{ijkl} = 1, i = 1, 2; j = 1, 2, \dots, n; l = 1, \dots, R_j \quad (2.9)$$

$$FT_{ijl} = ST_{ijl} + p_{ijl}, i = 1, 2; j = 1, 2, \dots, n; l = 0, 1, \dots, R_j, i > l \quad (2.10)$$

$$FT_{ijl} + Q(2 + y_{ijj'l'} - x_{ijkl} - x_{ijkl'}) \geq FT_{ij'l'} + p_{ijl}, \quad (2.11)$$

$$i = 1, 2; j = 1, 2, \dots, n; k = 1, 2, \dots, m_i, j < l$$

$$l = 0, 1, \dots, R_j, l' = 0, 1, \dots, R_j'; i > l$$

$$ST_{1jl+1} \geq F_{1jl} + p_{1jl+1}, j = 1, 2, \dots, n; l = 0, 1, \dots, R_j - 1 \quad (2.12)$$

Karimi and Dvoudpour (2014) proposed an algorithm, which uses Variable Neighborhood Search (VNS), for solving flow shop scheduling problems using Pareto solutions, which constitutes artificial chromosome and consider repetitive sequences for producing next generation offspring. The objectives were minimization of total weighted tardiness and make span. The algorithm executes a sequential mining procedure on the best-found solutions in the Pareto archive. The data mining technique search among the elite solution relationship and VNS act as guide to carry out search procedure in best direction. Crossover and mutation, operators produces offspring from parent generation and maintains diversification in the population, respectively. So mining of solutions along with VNS on the chromosomes increases the convergence and enhances the performance. When compared, the results of multi-objective genetic local search (MOGLS) algorithm were outperformed. The problem was coded in MATLAB was used for coding in 2.33 GHz Intel core 2 duo processor with 2 GB RAM.

Mirabi (2014) framed a novel hybrid genetic algorithm (HGA) which uses three genetic operators named as, order crossover, heuristic mutation and inversion mutation for large flow

shop scheduling problems including sequence dependent set-up times for every machine, to generate an improved population of chromosomes using an Iterated Swap Procedure (ISP), with an aim to minimize the makes-span. GA combined with modified NEH_RMB approach to produce initial population of chromosomes. The presented HGA compared to prediction error method (PEM), polynomial time heuristic (PH) and stochastic hybrid heuristic produced better results at a confidence level of 0.05. The problem was coded in MATLAB using Pentium III 1.2 Hz CPU with 512 MB RAM.

Navei et al. (2014) addressed the two-stage assembly flow-shop scheduling problem (TSAFSP), with sequence dependent set-up times, for second stages with the objective of minimizing the holding cost and delay costs. Simulated annealing (SA) and imperialistic competitive algorithm (ICA) were combined to determine a scheduling order of jobs at first stage and developed a heuristic (HEU) also. Then, four hybrid meta-heuristics were developed, for assigning the jobs at stage two of the assembly by combining SA and SA, SA and HEU, ICA and HEU, and SA and ICA. One-way analysis of variance along with computational results shows that combination of ICA and HEU outperformed all the three, with SA and HEU being the worst in terms of average error. The combination of SA and HEA gives the least computational time. Meta-heuristics were implemented in C++ 5.02 and 8.2 GHz processor with 2GB RAM is used.

Xiong et al. (2014) addressed a novel distributed two-stage assembly flow-shop problem (DTSAFSP), with objective to minimize the total completion time to assign the jobs to each factory. Three hybrid meta-heuristics proposed, namely HVNS, based on variable neighborhood search, HGA-RVNS, a combination of genetic algorithm and variable neighborhood search, and HDDE-RVNS, a combination of differential evolution and variable neighborhood search. ANOVA method was used to tune the parameters of HVNS and Taguchi method for the rest two. The analyses of computational results showed that for large number of jobs HGA-RVNS obtain better results and for small number of jobs HDDE-RVNS outperforms other two. The performances of both RVNS based algorithms were un-affected by the number of machine set-ups and RVNS based local search showed the effectiveness and efficiency of HGA-RVNS and HDDE- RVNS. The experiments were run on an Intel Core i7 3.4 GHz processor with 4 GB RAM.

Yu et al. (2014) developed a multi-objective optimization model for a two line-cell (seru) conversion assembly system with aim of optimizing the total labor hours (TLH) and total throughput time (TTT). Further, to find out characteristics such as its complexity, non-convex properties and solution space, this bi-objective model was examined. For solving large size problems in a reasonable computational time, presented a non-dominated sorting genetic algorithm. The paper clarified complexity and solution space of problem in which assembly line converted to pure cell system proved the non-convex nature of problem developed NSGA-II based algorithm and modified its operators to fit features of line-cell problem. In addition, compared the results with those obtained from enumeration methods. The problem was coded in C# in an Intel Core 2 processor of 3 GHz using 992 MB of RAM.

Fernandez-Viages and Framinan (2015) developed two algorithms namely, advanced non population based algorithm (ANPA) and a bounded insertion based constructive heuristic (BICH) by combining Tabu local search method and Taillard's acceleration for permutation flow shop scheduling problem with aim of minimizing the make-span and maximum tardiness, and constrained by two parameters, customer satisfaction and machine utilization. The combination of both the methods explored the solution space. The efficiency of both algorithms is measure of number of feasible solutions, number of instances with best solution and average relative percentage deviation. When compared to GA and FL algorithms, which are state-of-the-art-algorithms, newly developed proved to generate better solution and constituted the new state-of-art approximation solution procedures. The problem was coded in C# language in Intel Core i7 3.4 GHz processor with 16 GB RAM.

Gedik et al. (2015) proposed three formulations namely, logic based bendor decomposition algorithm, integer linear programming and constraint programming model, for unrelated parallel machine in a fixed planning horizon with an objective to schedule the non-similar jobs with sequence dependent set-up times, job availability intervals, non-identical job durations and unrelated machines. The jobs being constrained by minimum cost, maximum profit and assigned to only one machine in order to maximize the total profit. The study assumes cost and profit being independent of type of machine, soft transition gaps between constraints and environmental work window is incorporated by assigning different values to intensity function and master problems of decomposition algorithm are created for each iteration. The models provide feasible results when carried out on real life case study with

U.S. Army Corps of Engineers. IBM ILOG CPLEX Optimization studio was used to model the problem in Core 2 Duo @.93 GHz processor with 16 GB RAM.

Guo et al. (2015) developed a novel harmony search based multi-objective optimization model (HSMO), which is a combination of Pareto optimization method and Monte Carlo simulation process, for multi-site order scheduling problem under production un-certainties and, includes learning effects, in a make-to order manufacturing system. First, HSMO process was finds Pareto optimal solution to MMOP problem and then, Monte Carlo Simulation determines performance of each solution and manage production uncertainties. At-last, heuristic pruning process obtains final Pareto solution form the initial solutions. The optimum-seeking process was simplified by assigning production processes 1 of each order group. When compared, HSPA performs better than NSGA-II and hence is very effective in solving MMOP problems. Further, HSMO has a potential to solve stochastic multi-objective optimization problems.

Hatami et al. (2015) addressed distributed assembly permutation flow-shop scheduling problem with sequence dependent set up times with an objective of minimizing the makespan. There were two stages, production and assembly. In production stage, identical factories considered produces jobs in a flow-shop environment and then assembled to form final products through identical assembly program made by single machine. Set up times were included in both the stages. Two meta-heuristics and two simple heuristics were developed, and Design of Experiments (DOE), carried out for analysis. The algorithms such as Variable Neighborhood descent (VND) and Iterated greedy (IG), were proposed, calibrated, analyzed and further made simpler with addition of acceptance criterion with reduced parameters. Intel XEON 2.5 GHz processor with 16 GB RAM was used to solve the mathematical formulation of the heuristics.

Li et al. (2015) proposed a heuristic algorithm Minimum Attribute Ratio of Batch (MARB) for scheduling on single batch processing machine for minimization of tardiness and earliness of the jobs having a common due date. The problem considers non-identical job sizes, and introduced a concept named, attributed ratio of batch (ARB), which is the heuristic information to assign the jobs into batches and should be small as possible so to achieve the defined objective. Also, MARB combined with GA and a hybrid genetic algorithm was developed to study the problem, under the optimal properties, which were applied to

effectively schedule the batches. The study can be stretch to analyze parallel flow-shop environment or penalties can be included on earliness or tardiness. A mathematical model was formulated in IBM ILOG CPLEX and algorithms were coded in C# language on 2.2 GHz processor with 2 GB RAM.

Liao et al. (2015) formulated an efficient heuristic for scheduling of 'n' number of jobs in a two-stage assembly problem with set up times included for each job, in order to minimize the make-span. There were two levels, on first there existed a machine for machining the jobs, on second, a single assembly machine to assemble the product with similar components on the basis of purchase orders of clients. The proposed approach considered processing time and assembly time for each job. The set-up times were required at the start of the machining operation, or when parts were switched-on to machine. This problem is framed to be a mix-integer linear programming model (MIP), identified various properties to find optimal solution and a lower bound to check the performance of formed heuristic was derived. Two lower bounds based on machining and assembly time, were developed and higher value bound selected for further use in the approach. The software used for coding the heuristic was C++ on Pentium 2.51 GHz processor with 1 GB RAM.

Lin et al. (2015) developed Backtracking Search Algorithm (BSA), for permutation flow shop scheduling problem (PFSP) in the manufacturing industries with an objective to minimize make-span, and used simulated annealing mechanism and operators such as crossover, mutation to avoid random and premature local search. BSA combined with local searches to explore and exploit the search space developed a Hybrid backtracking search algorithm (HBSA). This algorithm utilized discrete crossover and mutation strategies along with SA to avoid falling into local optimal. HBSA produced global optimal solution and reduced the gap among the optimal solutions. Local search methods such as SA combined with meta-Lamarckian learning strategy, referred local search (RFL), pair-wise based local search were tried to solve PFSP problems and prove to be effective in producing more competent results. The algorithm coded in MATLAB 7.0 and simulated using Pentium dual core 3.0 GHz processor with 4 GB RAM.

Liou and Hsieh (2015) combined Genetic Algorithm and Particle Swarm Optimization to develop effective and efficient hybrid algorithm, for multi-stage flow shop group scheduling problem considering sequence dependent set-up times and transportation time, with an aim to

minimize the make-span. The algorithm determines sequence of groups and sequence of jobs in groups. Three lower bounds developed to verify quality of solution. The developed hybrid algorithm, which is fast and easy to configure, outperformed both GA and PSO in producing optimal solutions, particularly for the large-scale problems. The future scope of this algorithm could be finding solutions including minimizing weighted job tardiness, total tardiness and job blocking be considered. Problem, coded in MATLAB 7.0 used Intel-Pentium 2.4 GHz Processor with 1.97 GB RAM.

Pakzad-Moghaddam (2015) invented Levy Flight Embedded Particle swarm optimization (LFEPSO) algorithm, for scheduling jobs in parallel machine scheduling environment formulated as bi-objective mixed integer mathematical model. The objectives were to minimize maximum completion time and machine hiring cost. The algorithm can learn, adapt and depend upon the several parameters, which follow triangular distributions. PSO was modified to fit well in the complicated structure of problem along with levy flights, which were to replace the uniformly distributed walks and are effective in solving local optima problems of traditional PSO. LFEPSO outperformed exact solution method in scheduling 8-500 jobs on 3-50 parallel processors. When compared to PSO, LFEPSO managed to pass 20 % occasions where PSO stuck to find solution and gave better results but took more computational time. The numerical results showed undeniable gaps, after neglecting the adapting ability. This algorithm has a potential to solve problems with earliness, setup cost and tardiness included. The method uses Intel core i7-2640 M chip processor with 6 GB RAM.

Shahsavar et al. (2015) formulated three genetic algorithms (GA), named as two stage multi population genetic algorithm (MPGA), two phase subpopulation genetic algorithm (TPSGA) and non-dominated ranked genetic algorithm (NRGA), based on Pareto Optimal method, for project scheduling problems. The objectives were to minimize variability of resource usage known as resource leveling problem (RLP) which uses pattern of resource usage in the past time, reducing the resource usage cost known as resource investment problem (RIP) which assumes resources as unlimited and make decision on their availability level, and minimizing the make-span. NRGA outperformed TPSGA and MPGA according to the technique for order preference by similarity to ideal solution (TOPSIS). The algorithms were multi-

evolutionary and included two control strategies of self-adapting and five performance parameters such as variance of solution, diversity, convergence and extent of spread.

Smutnicki et al. (2015) developed Vector Evaluated Simulated Annealing (VESA) approach for solving multi-criteria flow shop scheduling problems, using fine grained parallel computing to explore the search space, which allows to approximate the Pareto front in competitive time known in other algorithms. VESA explores the solution space by considering cloud of individual solutions by using vector parallel processing in order to reduce computational time. By taking the advantage of multi-objective properties of problem, parallel-evaluated permutations were used to determine Pareto frontier. All the parallel-evaluated permutations Also, to verify the solutions, enhanced simulated annealing approach was formulated using the excellent approximation of Pareto front and produced 31% more results per unit time than SA. VESA used Intel Core i7 2.3 GHz processor to carry out the experiment.

Wang and Zhang (2015) presented heuristic algorithm for the flow-shop scheduling problem with set-up times included, which changes according to learning effects. The objectives were to minimize the weighted total make-span and total completion time. The processing time of a job was summation of the logarithms of jobs already processed and its position in the sequence. The presented algorithm was modified from the optimal schedules of the corresponding single machine-scheduling problem, so named as Modify-FL (MFL) and their worst-case error bound was analyzed. Also, branch and bound was adopted for m-machine permutation flow-shop problem. MFL has potential to study parallel machine problems, solve considering other objectives or consider genetic algorithm or Tabu search approach. The algorithm was coded in C++ 6.0 and 2.20 GHz processor with 4GB RAM was used to conduct mathematical experimentation.

Amini et al. (2016) addressed problem of truck scheduling in a cross-docking center, in which breakdown can happen during service time with aim to hand over the services before a pre-determined time i.e. a due date is assigned. Poisson distribution used to frame the breakdowns. There were two objectives, minimizing total tardiness and weighted completion time. A reliable system expected to determine whose breakdowns are zero and hence minimizing weighted completion time with same weights. Three meta-heuristics were formulated namely, Multi-objective Simulated Annealing (MOSA), Multi-objective

Differential Evolutionary (MODE) and Non-Dominated Sorting Genetic Algorithm (NSGA II). MOSA gave befitting results with lowest computational times and MODE proved to be the most efficient of the three. Factors associated with meta-heuristic were found using Response Surface Methodology (RSA). The algorithms were coded in MATLAB in Core 2 Duo 2.67 GHz processor with 4GB RAM.

Batur et al. (2016) presented a heuristic based on simulated annealing along with two neighborhood structures, for hybrid flexible flow shop environment with an aim to determine the best cycle time, which is constrained by robot movement, machine assignment and part sequences in a multi-part manufacturing system, where robots transport the parts between machines. There were two stages, first containing one machine and second with two machines. The lots with different parts were processed repeatedly and transportation is done by robot, hence considered as travelling salesman problem. The solution must define exact movement of robots to load, unload and carry the part in the system. Also, a lower bound was developed, which does not depend upon parameters such as number of stages, machines or parts and was outperformed by presented algorithm. The presented algorithm narrowed the gap between optimal solutions and was competitive to lower bound value. The algorithm was implemented in MS visual Studio 2010 using C++ on Pentium V 2.80 GHz processor with 1 GB RAM.

Martin et al. (2016) proposed an agent based framework in which each agent implements different meta-heuristic or local search method for permutation flow shop scheduling problems and capacitated vehicle routing, with an aim to provide a flexible framework, which is capable of dealing with various kinds of problem ranges. Ontology helps the agents to represent same internal structure. The agent adapts itself to the search process and improved solutions, identified by their frequency of occurrence. The solutions shared among agents and identified by cooperation protocol, which is a combination of reinforcement learning and pattern matching. Cooperation protocol means ability of parallel agents to share information in the whole process. Many agents had a confidence level of 95 % and those with somewhat less had a chance to improve. The framework needs very little tuning of parameters in the ongoing process. This study used Linux cluster with eight identical machines with two agents at a time, using a 2 GB memory.

Lin and Zhang (2016) developed Hybrid Biogeography Optimization Algorithm (HBBO) for minimizing the make-span in Distributed assembly permutation flow-shop scheduling problem (DAPFSP). There were two steps, first employing path-relinking heuristic in migration stage as local search strategy, and second being insertion based heuristic to determine job permutation in migration stage. The local search presented explores the solution space. BBO designed includes many efficient heuristics and evaluated by comparing with two sets of benchmark instances. The new optimum solutions found in 71 small size instances and 91 large-size instances. Further, fitness landscape analysis applied to explore characteristics of solution space, hence reducing the computational time. BBO being a high-level strategy, applied for low-level heuristics for optimal scheduling in DAPFSP. The algorithm was coded in C++ language using Intel Core I3 2.50 GHz processor with 4 GB RAM.

Liu et al. (2016) introduced multi-objective genetic algorithm (GA) based on non-dominated sorting genetic algorithm (NSGA-II) with additional two steps to expand the solution space, for minimizing total weighted tardiness and non-processing electricity consumption giving the schedules of machining without turn off/on the machines repeatedly, which consumes the electricity. The proposed algorithm is an intelligent scheduling method, which combines fragmented short idle periods on machines into large ones to reduce energy consumption. Local search Heuristic and shifting bottleneck heuristic were used to deliver the baseline scenarios for the machines. The Pareto fronts produced gives optimal solutions for scheduling, and simultaneously carrying out production and switching off underutilized resources. The performance testing has been done including the electrical profiles of the machine tools. When compared with NSGA-II produced better results in reducing non-processing electricity consumption along with minimizing tardiness.

Saraswat et al. (2016) presented a model using simulated annealing approach for planning block layout design with an objective of minimizing the average work-in process, minimizing the flow distance and number of material handling devices required, constrained by no empty material transfer, which increases equipment cost and work-in-process. The model considered accurate model for flow-distance optimization, sequence pair representation and work in process calculator to determine average work in process layout. The research neglected intra-departmental queues, which requires exact number of machines

in each department. Since the departments may have different or similar input and output stations, also presented a procedure to determine the input and output station of each department. The problem computed on Pentium IV 3.2 GHz processor with 1GB RAM using ILOG OPL Studio 3.7 and ILOG CPLEX 9.0, and coded in C++ 6.0.

Velez-Gallego et al. (2016), presented a novel mixed integer linear program (MILP) to schedule the set of jobs, in manufacturing processes like metalworking or painting, constrained by release dates and sequence dependent set-up times with an aim to minimize the maximum completion time of all jobs. A beam search algorithm was developed to find the optimal high quality solutions at low computational costs in reasonable computational time with no effect of release dates. The proposed model can solve small to medium sized instances and varying release dates affects the performance of MILP. Intel Xeon 8 core 2.337 GHz processor with 16 GB memory, used to run the experiment in Rocks 6.1.1 Linux distribution.

Khorasanian and Moslehi (2017) presented variable neighborhood search algorithm (VNS) for two-machine flow shop scheduling problems with an aim to minimize the make-span and constrained by blocking. Two characteristics, multi-task flexibility to process the operations of at least one other machine a preemption, to allow the solution space to grow in order to find efficient solutions, were added. As schedules were infinite in number, a dominant schedule was defined for each sequence. A variant of VNS called dynamic VNS (DVNS) presented produces high quality solutions for large sized instances and two mathematical models formulated for small sized instances. VNS algorithm was coded in C# and GAMS 24.4.6/CPLEX solved the mathematical models using Intel core 3.06 GHz processor with 4GB RAM.

CHAPTER 3: PRESENT WORK

3.1 INTRODUCTION

The flow-shop scheduling environment consists of 'n' number of jobs, to be processed on 'm' number of machines, following the same sequence. The foremost objective of flow-shop scheduling is to arrange the jobs of the manufacturing system to get optimized or maximum productivity and hence, utilizing all the resources (man, machinery, finance). In the previous sections the extensive overview described the various methods to solve the flow-shop problem under different objectives, parameters and constraints. In view of this, the most common issue is to optimize the make-span of the production system. Further, lateness, tardiness (or weighted lateness and weighted tardiness) with a sequence dependent set-up times and due dates are the secondary issues taken in consideration. As there can be infinite ways to arrange jobs, selecting an optimum schedule for the jobs on each machine and their execution at the time allocated will complete the orders before the due dates. Dispatching to the required locations with well marketing can exponentially increase the sales, maintains customer loyalty and hence the profit, which is the ultimate aim of any industry. Tardiness and lateness employ to the underutilization of resources, results in problem of backlogging too. The problems with backlogging, sequence dependent set-up times and due dates have been enlightened in the recent years because of the presentation of new techniques for stock administration, for example, just in time (JIT) manufacturing system which does not allow a moment to be spared. It is to be ensured that jobs are finished neither too soon nor past the estimated time which arises the scheduling issues with both earliness and lateness costs. The capacity of the system is also another parameter to be considered and to be improved. The amount of stock to be streamlined and the requirements go hand in hand. If more stock is generated then it will cause backlogging, and if it is less than decreases the efficiency of the production system. Nowadays, all the parameters are used as issued a term weighted as a predecessor, which attempts to calculate the exact need or importance of the parameter. More is the weightiness more will be the priority given to fulfil that criterion among others. In today's competitive era, the expense of creation must be lessened keeping in mind the end goal to get by in this dynamic environment which is finished by viable use of the considerable number of assets and fulfilment of generation in shorter time to expand the efficiency with at the same time considering due dates of the job. It is to be kept in mind that optimization of makes-span should be with respect to assigned due dates, otherwise it will

be of no use, as orders will not be delivered on time, intern causes loss of market for the product. Therefore, in present day fabricating environment industry needs to overcome every issue in order to stay into the clashing cum negotiating markets. Hence, keeping in mind the end aim to maximize the profit and market space for the company, there is a need of multi-objective scheduling framework, which is capable of accomplishing each and every aspect of the system simultaneously and in the specified time. So, considering the realistic scenario, this present work tries to manage flow-shop scheduling problems for optimization of make-span. This can be considered as a basic goal to accomplish utilization of assets in admiration of increasing the effectiveness and expanding the efficiency, meeting the due dates so, as to gain more customer satisfaction with improving the brand name.

3.2 PROBLEM FORMULATION

The flow-shop scheduling problems are regarded as non-deterministic polynomial (NP-Hard) time problems, whose exact solutions are difficult to find due their complexity and takes a significant amount of time. In the past years, various methods have been proposed such as genetic algorithm (GA), simulated annealing (SA), immune algorithm (IA), ant colony optimization(ACO), branch and bound (B&B), particle swarm optimization (PSO), tabu search (TS) and differential evolution (DE). These heuristics are used alone or can be combined with one another making a hybrid heuristics. Further, some search methods like local search technique, variable neighbourhood approach are applied to explore the search space and chose the best among the solution. All these heuristics are used to achieve the desired objective with a reasonable computational time. The problem can be formulated by dispatching the rules, constructing the heuristic and improving heuristic. Dispatching of the rules will initiate the formulation process by building the initial schedule for the further process. A series of passes is made through the unscheduled jobs in constructive heuristics, which adds one or more jobs in the schedule. Improvement heuristics is a reverse process as it starts from a convenient solution and tries to improve it. The parameters, assumptions, constraints, objectives and collected relevant data should be well defined before initialising the mathematical computation in the software's. The software used for coding is selected so it is compatible, fast, and reliable with respect to the algorithm. Some frequently used software's are LINGO, MATLAB and languages for coding are C, C# and C++.

3.2.1 Assumptions

The assumptions used in the flow-shop scheduling problems are as follows:

- (i) Jobs are independent and are available for processing at time zero.
- (ii) All the descriptors regarding the respective job are known before starting any operation on it.
- (iii) At least one machine is available at all the time.
- (iv) No machine is kept idle.
- (v) A job will be passed onto next machine only after its completion.
- (vi) Pre-emption is not allowed.
- (vii) Machines are accessible all through the scheduling period.
- (viii) Each machine is ceaselessly accessible for task, without critical division of the scale into movements or days and without thought of provisional inaccessibility, for example, breakdown or support.
- (ix) The system may have movable machines.
- (x) In-process stock is permitted. In the event that the following machine on the arrangement required by a job is not accessible, the job can hold up and joins the line at that machine.

3.2.2 Considerations

There are 'n' number of jobs to be scheduled in a specific order in a flow-shop machine arrangement in order to optimize the objectives. The jobs follows the constraints presented below:

- (i) Set-up times are attached with each job
- (ii) No-wait constraints.
- (iii) Multiple criteria's are to be optimized.
- (iv) At a given time, no two jobs are processed on a machine.
- (v) Two machines cannot process the same job at the same time.

3.3 RESEARCH GAPS

- (i) Implementation of the heuristics methods by combination and cross-functioning of performance measures of scheduling such as tardiness, lateness, due dates, minimization of make-span, considering sequence dependent set up times and backlogging have not been executed.
- (ii) The retrospection of the research aims to utilize the conventional methods designed decades back such as GA, PSO, SA, and AI and so on, which restricts the development of

the newly formed methods to solve the flow shop scheduling problems. Further, the examination of the considered constraints namely, sequence dependent setups and backlogging have been seen in fewer studies with the non-conventional methods, hence widening the scope of more work to be implemented in future.

(iii) Fewer case studies which are based on the real data analysis of the various parameters such as process time, earliness, tardiness and others, have been taken less into consideration related to multi-objective flow shop scheduling problems and its constraints. The previous research lacks more realistic formulations, which can be reverted, back to improve the respective system in the industry.

(iv) Investigation of a hybrid scheduling problem where the machine configurations such as open; permutation; flexible, and hybrid are more complex and the integrated problems as formation of multiple scheduling are even harder to solve as it includes various parameters of production, inventory, distribution, total cost and service level, which have not been contributed in the earlier works.

(v) The scheduler faces difficulty in selection of the appropriate algorithm for flow-shop scheduling under specified variables and objectives as cross-validation of these algorithm methods is less contributed. The computational results of various heuristics should be compared for the given problem in the validated data sets.

(vi) Lesser attempts have been conducted to develop some dominance conditions based upon data identification that can either be independent of schedules of the previous job or schedules with lesser number of jobs to be rejected quickly.

3.4 OBJECTIVES

(i) To investigate a hybrid scheduling problem with complex machine configurations with multi-objectives. Since the multi-objective flow shop problem (MFSP) is a special case of flow shop scheduling problem termed as NP-hard as it includes more than one objective to be solved.

(ii) To formulate a heuristic method to minimize the process time and cycle time for flow shop scheduling problem related to sequence dependent setup time with each part under full backlogging consideration.

(iii) To simulate a novel heuristic algorithm for solving the multi-objective flow shop scheduling problem with stochastic parameters by applying heuristic approaches such as tabu search, branch and bound, genetic algorithm, simulated annealing, ant colony optimization other heuristic methods, preferably latest optimized approaches such as

Teacher-Learning Based Optimization (TLBO) and Biogeography-Based Optimization (BBO) algorithm in proposed research work.

(iv) To formulate and implement the novel heuristic method by considering performance measures of production scheduling, such as process time and cycle time criterion and respective optimization.

(v) To validate the proposed research model by the comparison of analytical results generated by the implementation of novel hybrid heuristic algorithm approach in MATLAB with the optimized results generated by mathematical empirical relations from hybrid BBO-TLBO heuristics approach. This validation will also address the superiority of the proposed algorithm and further, the validated proposed best-fitted model will present the possibility of an improved manufacturing system with increased production efficiency.

3.5 OBJECTIVE FUNCTION

The following section aims at minimization of the dual objectives for flow shop scheduling problem, namely process time and cycle time by modelling a novel hybrid heuristic calculation. For this, a case with 8 machines and 2 parts is considered in which 2 parts are processed by the 8 different machines in a way such that it spends least time for processing of all the parts. The 8 machines and 2 parts are assumed as: -

Table 3.1: Model Problem of Flow Shop Scheduling with 2 parts and 8 Machines

Parts /Machines	M1	M 2	M 3	M 4	M 5	M 6	M 7	M 8
Part 1	J ₁₁	J ₁₂	J ₁₃	J ₁₄	J ₁₅	J ₁₆	J ₁₇	J ₁₈
Part 2	J ₂₁	J ₂₂	J ₂₃	J ₂₄	J ₂₅	J ₂₆	J ₂₇	J ₂₈

The Mathematical model of the problem is as follows:

J_{MN} – Part M processing on Machine N.

$M = \{1,2\}$ and $N = \{1,2,3,4,5,6,7,8\}$

Let Process – Z and Cycle Time – C.

Objective Functions are Minimize (Process Time [Z]) and Minimize (Cycle Time [C]).

The Constraints are mentioned below as:

(i) Only one machine should be selected from the set of available machines for each operation.

- (ii) The preventive maintenance tasks have to be executed within their time windows.
- (iii) The feasibility of the two variables.

3.6 HEURISTIC METHODS AND HYBRID ALGORITHM

The two heuristics which will be utilized to optimize the current problem are Teacher Based Learning Optimization (TLBO) and Biogeography Based Optimization (BBO). The first two subsections below will enlighten about the each of the algorithm respectively, and the third will be devoted to the formulation of novel hybrid algorithm by using the features of above two.

3.6.1 Teacher Learning Based Optimization (TLBO)

The efficient optimization method named TLBO was proposed by Rao et al. in 2011 which is based on the teacher and student learning process. It is a naturally inspired population method, where class of learners will represent the population. The best learner in the process is selected as a teacher, as only a teacher is considered with best knowledge and increments the knowledge level of the students known as learners, so as to obtain the good marks. Here, the capability of a teacher to deliver and the quality of the class present also plays an important factor in order to increase the average of the class. The mean value of the class indicates the quality of class. After reaching a certain level of knowledge, the class needs a new teacher who can impart a better content with increased capability. The best learner in the process is selected as a teacher, as only a teacher is considered the best knowledgeable person in the class. There are two phases which constitutes the whole process namely, teacher's phase i.e. grabbing knowledge directly from the teacher and learner's phase, which motivates the grabbing knowledge between the learners. In the teacher phase, the teacher approaches to impart all of his knowledge among the class which is impractical in reality. This is because of the difference in the capability of delivering by teacher and that of understanding by the students. The teacher efforts to move the mean value to its own level. The learner phase on the other hand, inputs the knowledge from teaching phase and then further, increases it by interaction among the learners. The learners interacts randomly with one another through presentations, discussions or formal communication. But the level of learner can only increase if the other learner has more knowledge, so here the job of the teacher is shifted to the more knowledgeable individual, who now tries to increase knowledge level of the former their own respective levels.

3.6.1.1 Steps in the implementation of TLBO

There are number of steps involved in the formulation of the TLBO algorithm consisting of two phases, namely teacher and learner.

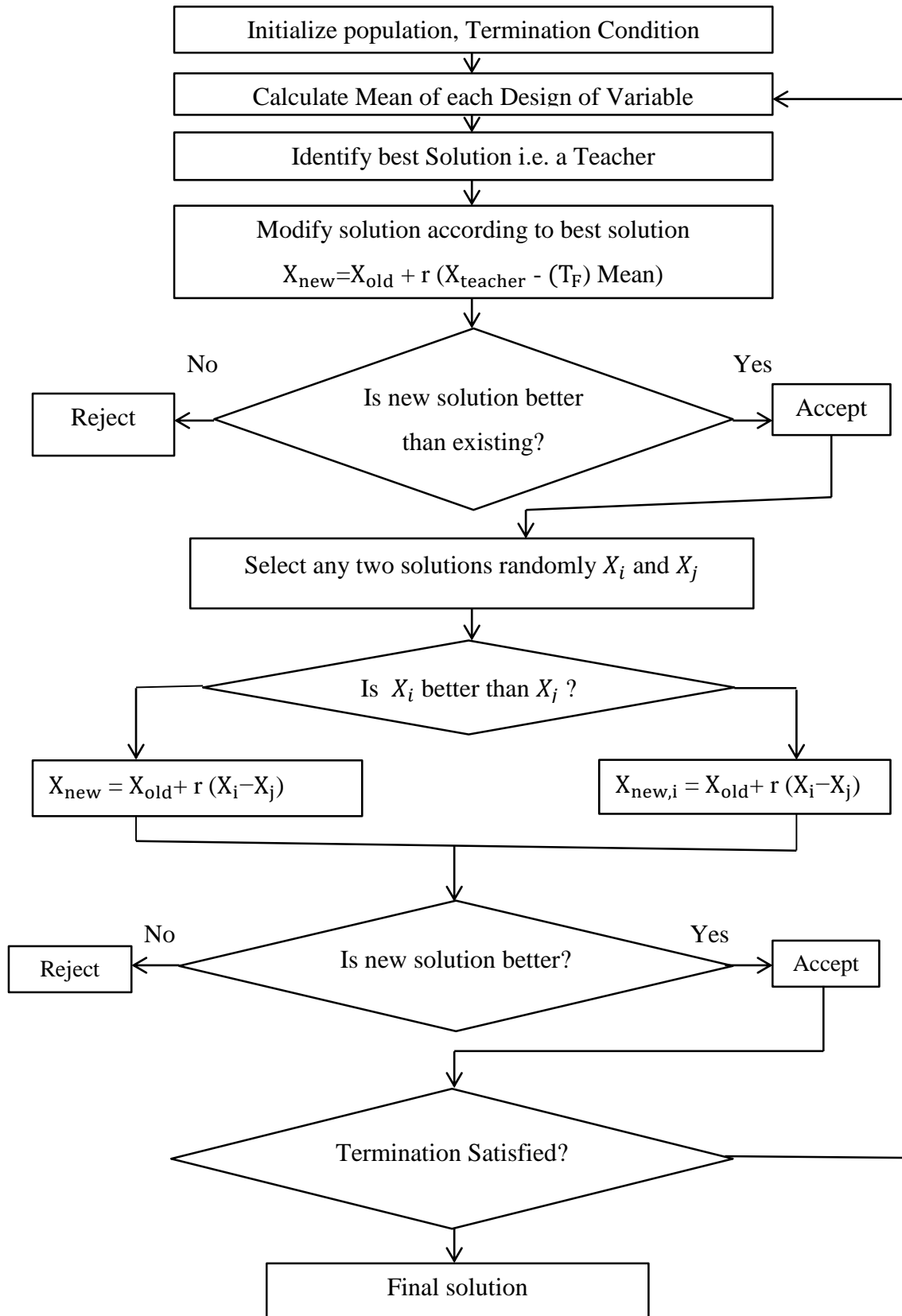


Figure 3.1: Depiction of Process of TLBO for Optimization of MFSP

The steps include notations, equations and description of each step. The following are the steps that are used in the implementation of TLBO:

Step 1: Defining of the optimization function and initializing the parameters such as population size (P_n), generations to be computed (G_n), variables (D_n), upper bound (U_1) and lower bound (L_1). Minimize $f(Z)$, where $Z \in 1, 2, \dots, D_n$.

$F(Z)$ = objective function and Z is a vector for design variables such as $L_{L,i} \leq Z_i \leq U_{L,i}$.

Step 2: Initialization of Population

A random set of population is generated depending upon the population size which represents the learners and decision variables represent the subjects offered.

Step 3: Teacher Phase

The column wise mean of the population is calculated giving the mean of every subject such as $M_{,D} = [m_1, m_2, \dots, m_D]$

The best solution will act as teacher for iterations $X_{teacher} = X_{f(X)=min}$

Now, the teacher efforts to increase mean from $M_{,D}$ to $X_{teacher}$ by $M_{new,D} = X_{teacher,D}$

The difference between two means is $Difference_{,D} = r(M_{new,D} - T_F M_{,D})$, where T_F is the teaching factor 1 or 2.

The current solution is updated by $X_{new,D} = X_{old,D} + Difference_{,D}$

Step 4: Learner Phase

The learners increase their knowledge by interaction and given by

For $i = 1:P_n$

Randomly select two learners X_i and X_j , where $i \neq j$

If $f(X_i) < f(X_j)$

$X_{new,i} = X_{old,i} + r_i(X_i - X_j)$

Else

$X_{old,i} = X_{old,i} + r_i(X_j - X_i)$

End If

End For

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

Step 5: Termination Condition

Stop when the maximum generation number is reached; otherwise iterate step 3.

3.6.2 Biogeography Based Optimization (BBO)

Biogeography is inspired from the nature's geographic dispersion and proportioning of the biological organisms and was formulated by Dan Simon in 2008. It imbibe features of

genetic algorithms and particle swarm optimization therefore can be utilized for the same problems these two. The field of biogeography was studied by Alfred Wallace [1] and Charles Darwin [2] but the mathematical formulations were framed by Robert MacArthur and Edward Wilson in 1960's. BBO is capable of laying down the mathematical models for migration of the species and their extinction along with the rise of new species. This is done in order to relocate the population of species to the neighbouring islands. The term island refers to the habitat which has been isolated from the other habitats. The geographical areas are home to the species and are affected by natural conditions such as rainfall, temperature, topography and vegetation. For population to grow, and it is supposed to have high suitability index (HSI) which is dependent on the natural conditions whereas suitability variable index is independent of the conditions. High HSI will lead to emigration of various species to the nearby habitats by virtue of large species they host. The immigration rate will be very less due to already existence of saturated species. Low HSI habitat experiences high immigration rate due to their sparse population and results in increase of the HSI. But, if the HSI remains low, the species will tend to extinct. A good or bad solution is proportionate the high or low HSI value, respectively. The low and high HSI habitats share the features that remain in high HIS, and new features are observed in low HSI habitat. The quality of the solutions is increased due to the formation of new features in low HSI as it has the ability to accept changes in the habitat, as compared to the high HSI habitat which resists changes.

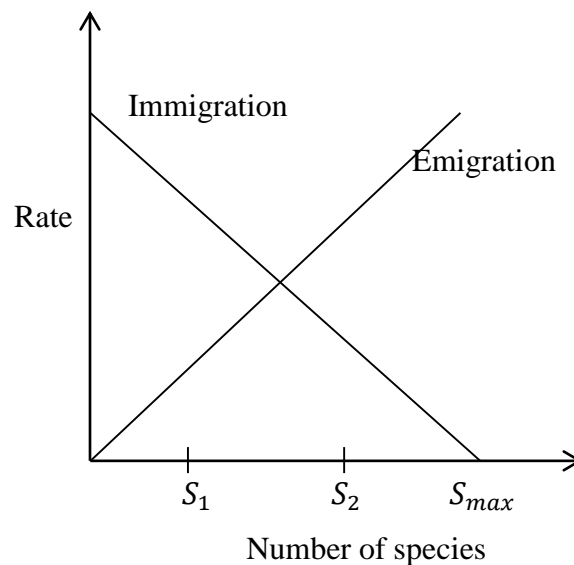


Figure 3.2: Depiction of Relation between Emigrations, Immigration Rates w.r.t. Number of Species for BBO Algorithm

The maximum immigration can take place when there is no population in the habitat. With the increase in population, the crowdedness rises and it becomes difficult to survive with the immigrants hence results in the decrease of the immigration rate. In case of emigration, with the rise in the population, now the species has the opportunity to discover new habitats for residence purposes, hence the emigration rate increases. BBO utilizes two operators namely migration and mutation. The migration resembles the other evolutionary methods, in which parent produces an off-spring with a little distinction in the features. The migration is an adaptive strategy which alters the existing solutions. Elitism is also used along with migration to store the best solutions without any corruption due to immigration.

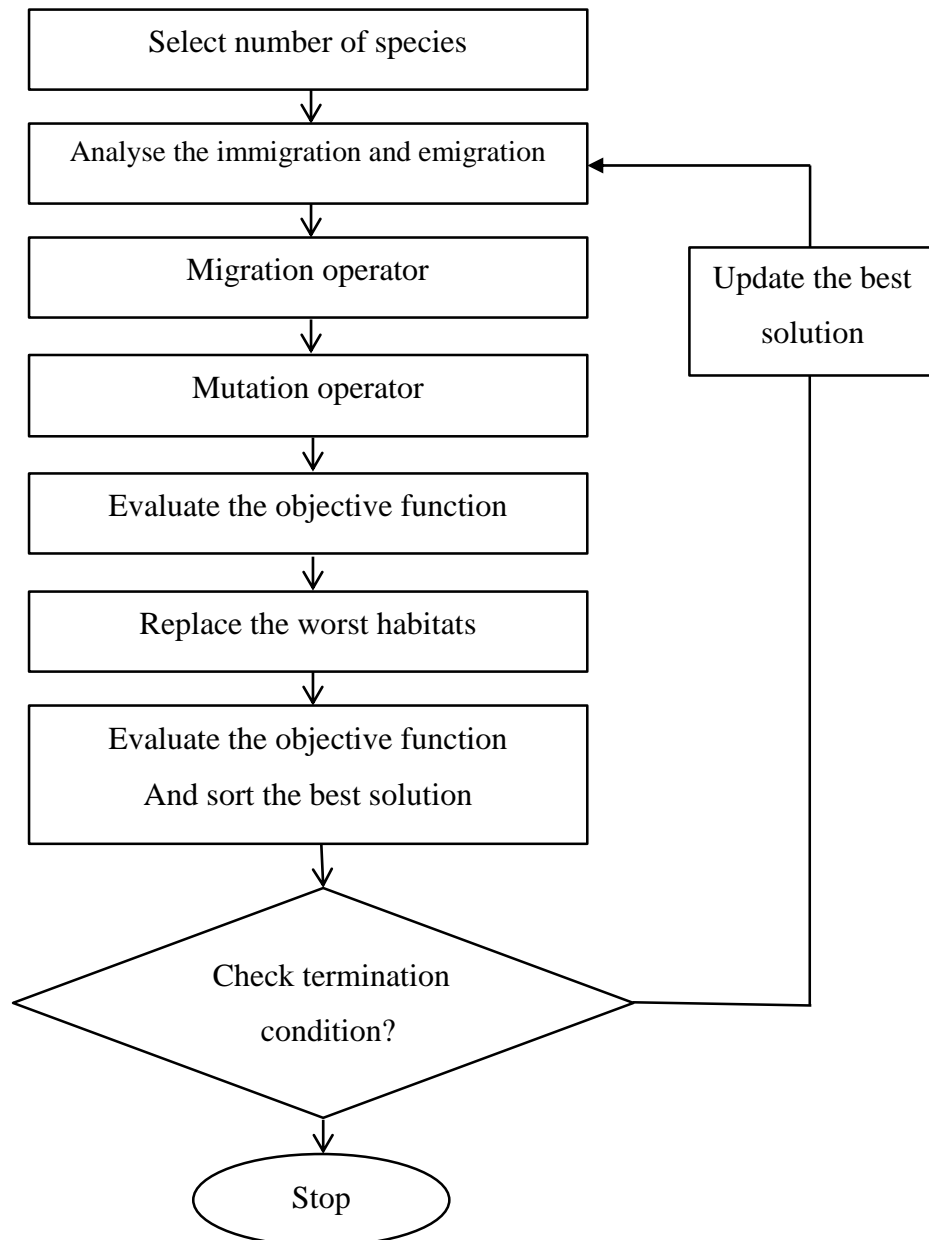


Figure 3.3: Depiction of Various Steps Included in BBO for Optimization of MFSP

Mutation rates are determined by the probabilities of the species count. Every member of the species is associated with a probability and medium HSI values are considered to be probable than high or low HSI. If a low HSI exist, it will tend to mutate to the other solution and that with high probability resists the mutation to the other solution. Mutation raises the diversity among the species and is inversely proportionate to the probability. There is a possibility to improve the solutions by mutating them and uses elitism to revert back the best solution. The steps involved in the implementation of BBO are mentioned as:

Step 1: Initialize the BBO parameters such as maximum species count, the maximum migration rates, the maximum mutation rate, and an elitism parameter. The maximum species count and the maximum migration rates are relative quantities. That is, if they all change by the same percentage, then the behavior of BBO will not change. This is because if and change, then the migration rates and the species count will change by the same relative amount for each solution.

Step 2: Initialize a random set of habitats, each habitat corresponding to a potential solution to the given problem.

Step 3: For each habitat, map the HSI to the number of species, the immigration rate, and the emigration rate.

Step 4: Probabilistically use immigration and emigration to modify each non-elite habitat and then re-compute each HSI

Step 5: For each habitat, update the probability of its species count using. Then, mutate each non-elite habitat based on its probability, and re-compute each HSI.

Step 6: Go to step (3) for the next iteration. This loop can be terminated after a predefined number of generations, or after an acceptable problem solution has been found.

3.6.3 Novel Hybrid Heuristic: TLBO's Teacher Phase Based BBO (BBO-TLBO)

The two methods which are utilized to solve the respective problem are Teacher Learning based Optimization (TLBO) and Biogeography Based Optimization (BBO), which are described in the above subsections as well. Where TLBO undertakes the learning mechanism or process between student and teacher as their foundation to solve the optimization problems, there on the other hand BBO takes its inspiration from the distribution of natural habitats. The execution of BBO is similar to the other evolutionary heuristics such as GA and PSO whereas TLBO depends on the capability of teacher to deliver and that of the learner to adapt. The two phases that exist in the TLBO are named as teacher phase and learning phase. The teacher phase proclaims the teachers to share

and impart the complete knowledge that they possess among the learners. On the contrary the learner phase encourages the sharing of knowledge among the group of learners to gain the same level of knowledge. The teacher phase here is collaborated with the BBO's working mechanism and is added to the migration section in the BBO mechanism. Migration has the ability to improve the solutions by altering the solutions produced. So, in context to produce the solutions the teacher phase is inserted, hence affirming the new solutions. The researches till now were contrary to this step, as they utilized the mutation or migration phase of BBO in the TLBO working mechanism. There are number of steps for the execution of the proposed hybrid heuristic, enumerated as below:

Step 1: Defining the problem

The objective function of the problems is to be defined.

- (i) Define the objective function: $Objfun(x)$. The objective function contains the equations formed from the system.
- (ii) Define the number of variables in the objective function: $nVar$. All the unknowns in the objective functions on whose value changes during the working of algorithm are mentioned.

The screenshot shows the MATLAB Editor window for a file named 'TBmain.m'. The code is as follows:

```

1  %% Problem Definition
2
3  CostFunction=@(x) expensive_TB1(x);    % Cost Function
4
5  nVar=2;                               % Number of Decision Variables
6
7  VarSize=[1 nVar];                    % Decision Variables Matrix Size
8
9  VarMin= 1;                            % Decision Variables Lower Bound
10
11 VarMax= 15;                           % Decision Variables Upper Bound
12

```

Figure 3.4: Depiction of the Step 1 which Defines the Problem in MATLAB Software

(iii) Define the other lower bound: VarMin. This the lower value for the variables which they can achieve.

(iv) Define the upper bound: VarMax. This is the upper value for the variables which they can achieve.

The lower and the upper bounds is the range for the variables, within which the values for the variable will be roaming to satisfy the objective function.

Step 2: Describing the Parameters of the Heuristic

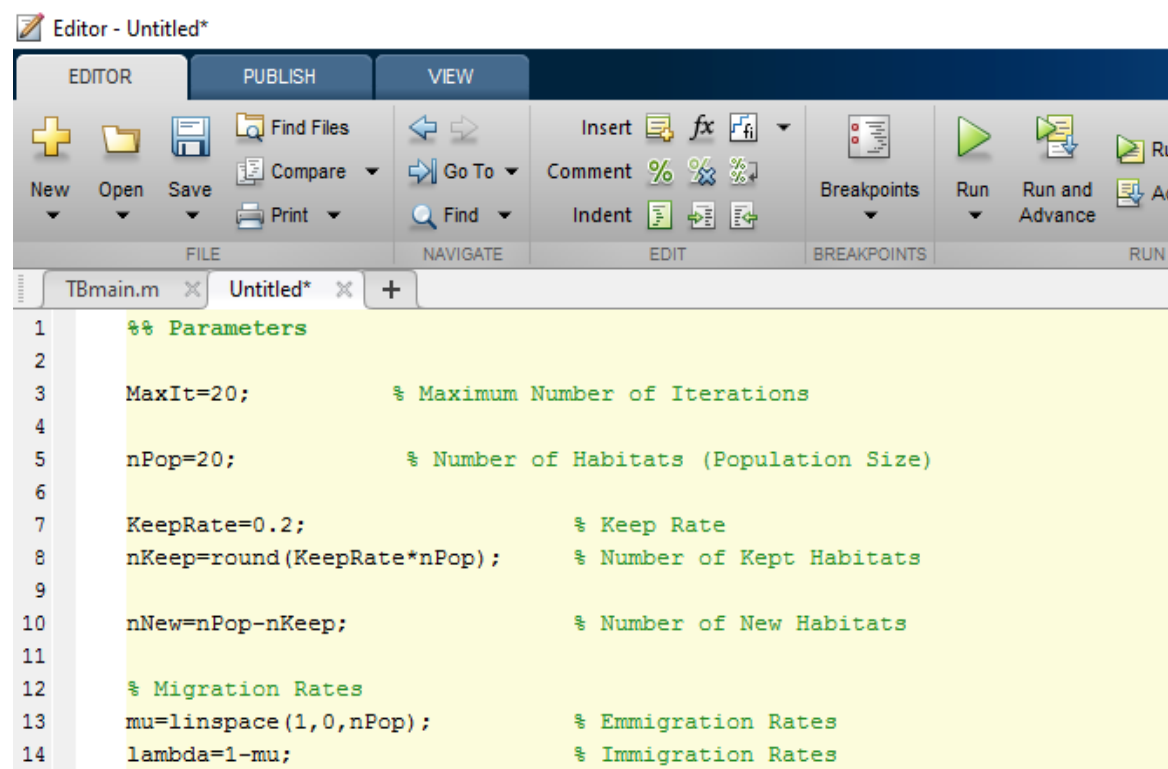
(i) Define the number of iterations to be executed: MaxIt. It is to be noted that, this will serve as our termination condition and the heuristic will stop working after the specified number of iterations.

(ii) Define the population size of the habitat: nPop. This is the maximum number of species residing in the habitat.

(iii) Define the immigration rate: mu. This is the rate at which the population can immigrate to the other habitats.

(iv) Define the emigration rate: lambda. This the rate at which the population from a specific habitat can exit to the other habitat.

(v) Define the Mutation rate: pMutation. This the rate at which the population in the habitat will be mutated.



The screenshot shows the MATLAB Editor interface with a menu bar (EDITOR, PUBLISH, VIEW) and a toolbar. The code in the editor is as follows:

```
1 %% Parameters
2
3 MaxIt=20;           % Maximum Number of Iterations
4
5 nPop=20;           % Number of Habitats (Population Size)
6
7 KeepRate=0.2;      % Keep Rate
8 nKeep=round(KeepRate*nPop); % Number of Kept Habitats
9
10 nNew=nPop-nKeep;  % Number of New Habitats
11
12 % Migration Rates
13 mu=linspace(1,0,nPop); % Emigration Rates
14 lambda=1-mu;      % Immigration Rates
```

Figure 3.5: Depiction of Step 2, which Defines the Parameters in MATLAB Software

Step 3. Initialization of the process

- (i) Create the empty arrays for the population to reside.
 - (a) habitat.position, will define the position of the specie.
 - (b) habitat.cost, will define the value of function w.r.t position of the specie.
 - (ii) Intialize the habitats.
- for i=1:nPop

```
pop (i).Position=unifrnd (VarMin, VarMax, VarSize);  
pop (i).Cost=Cost Function (pop (i).Position);
```

end

Here, unifrnd, is a value which will be randomly selected by the algorithm to initialize the population but in correspondence to the parameters such as lower bound, upper bound.

(iii) Sort the population: pop=pop (SortOrder). The population of the species is sorted in particular order which depends on the cost value in the decreasing order.

(iv) Create an array to hold the best cost: BestCost = zeros (MaxIt, 1). The best cost among the iterations can be separately viewed by this function.

Step 4: Formulation of the loop containing both the algorithms

```
%% Initialization  
  
% Empty Habitat  
habitat.Position=[];  
habitat.Cost=[];  
% Empty Structure for Individuals  
empty_individual.Position = [];  
  
% Create Habitats Array  
pop= repmat (habitat,nPop,1);  
  
% Initialize Habitats  
for i=1:nPop  
    pop (i) .Position=unifrnd (VarMin,VarMax,VarSize) ;  
    pop (i) .Cost=CostFunction (pop (i) .Position) ;  
end  
  
% Sort Population  
[~, SortOrder]=sort ([pop.Cost]);  
pop=pop (SortOrder) ;  
  
% Best Solution Ever Found  
BestSol=pop (1) ;  
  
% Array to Hold Best Costs  
BestCost=zeros (MaxIt,1) ;
```

Figure 3.6: Depiction of Step 3 which Initializes the Algorithm in MATLAB Software

(i) Start the loop by using for loop, which will be repeated for the population size mentioned above.

```
for it=1:MaxIt
```

```
    newpop =pop;
```

```
    for i=1:nPop
```

(ii) Start the Migration loop inside the main loop above.

```
for k=1:nVar
```

```
    if rand <= lambda (i)
```

(a) Here the TLBO teacher phase starts. Calculate the mean of the population.

```
        Mean = 0;
```

```
        for i=1:nPop
```

```
            Mean = Mean + pop (i).Position;
```

```
        end
```

```
        Mean = Mean / nPop;
```

(b) Selection of the best teacher modified to the best habitat.

```
            habitat = pop (1);
```

```
            for i=2:nPop
```

```
                if pop (i).Cost < habitat.Cost
```

```
                    habitat = pop (i);
```

```
                end
```

```
            end
```

(c) Create the empty solution

```
                for i=1:nPop
```

```
                    newsol = habitat;
```

(d) Define the teacher factor modified as habitat Factor

```
                    HF = randi([1 2]);
```

(e) Define the function to move towards the best solution

```
                    newsol.Position = pop(i).Position ...
```

```
                        + rand(VarSize).*(habitat.Position - HF*Mean);
```

```
                End
```

```
End
```

```

% Select habitat
habitat = pop(1);
for i=2:nPop
    if pop(i).Cost < habitat.Cost
        habitat = pop(i);
    end
end

% Teacher Phase
for i=1:nPop
% Create Empty Solution
newsol = empty_individual;

% Habitat Factor
HF = randi([1 2]);

% Habitat (moving towards habitat)
newsol.Position = pop(i).Position ...
+ rand(VarSize).*(habitat.Position - HF*Mean);
end

```

Figure 3.7: Depiction of Teacher Phase in Proposed Heuristic in MATLAB Software

(iii) Now, the Mutation phase of the BBO starts.

```

if rand<=pMutation
    newpop(i).Position(k)=newpop(i).Position(k)+sigma*randn;
end
end

```

(a) Apply lower and upper bound limits

```

Newpop (i).Position = max (newpop(i).Position, VarMin);
Newpop (i).Position = min (newpop(i).Position, VarMax);

```

(b) Evaluation the new cost function

```

Newpop (i).Cost = Cost Function (newpop(i).Position);
end

```

(iv) Sort the new population

```

[~, SortOrder]=sort ([newpop.Cost]);
Newpop = newpop (SortOrder);

```

(v) Select next iteration population

```

Pop = [pop(1:nKeep)
newpop (1:nNew)];

```

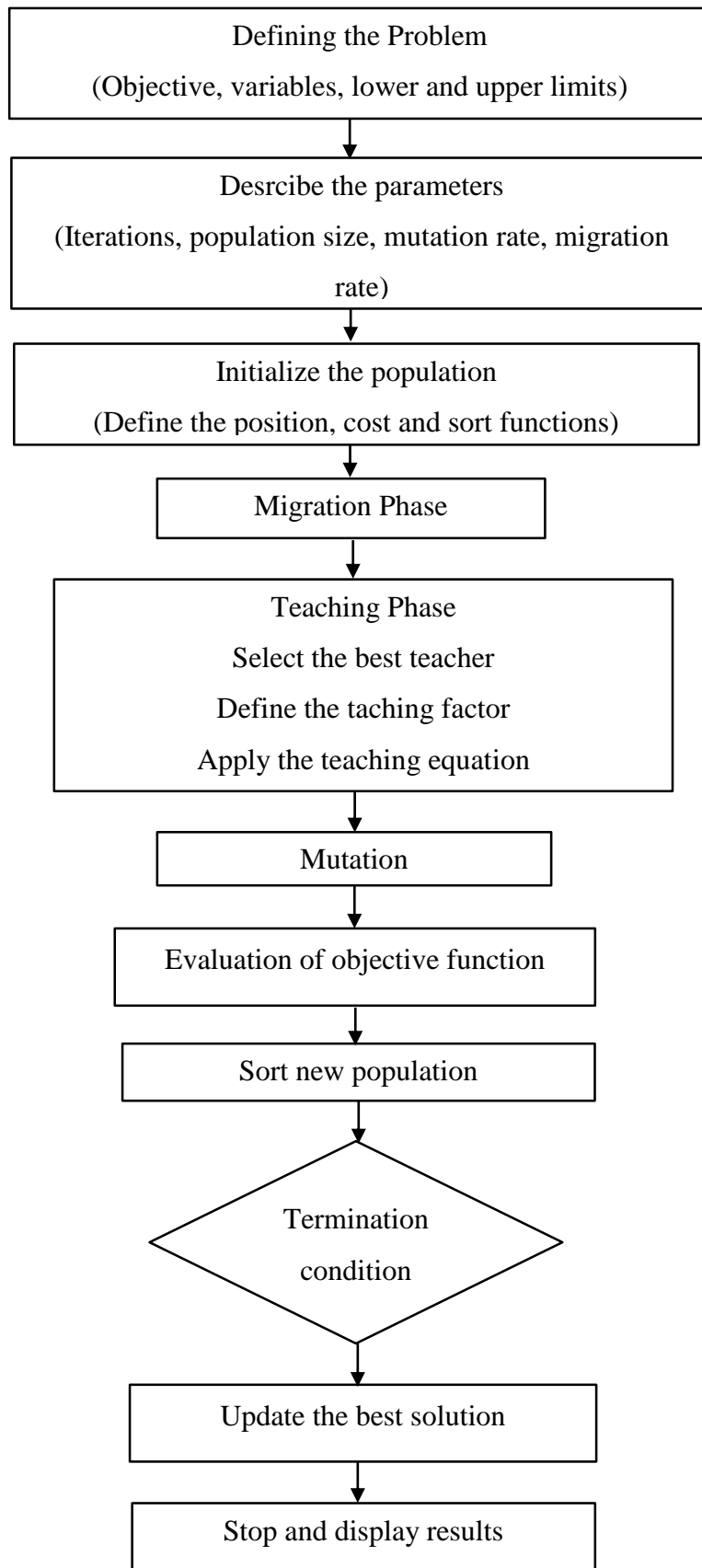


Figure 3.8: Depiction of Various Steps of Proposed Hybrid Heuristic for optimization of MFSP

The proposed heuristic which includes the teacher phase of TLBO algorithm in the migration and mutation phases of the BBO algorithm has various steps, as coded above. The steps include defining the objective functions, defining the parameters, migration phase, teacher phase, mutation phase, sorting the newly formed population and checking the termination condition. All these steps are necessary and make the proposed heuristic successful in its objective of producing the better results. The steps are depicted flow chart below in Figure 3.8.

3.7 COLLECTION OF DATA FROM INDUSTRY TO SOLVE MFSP

Federal Mogul is a piston ring manufacturing industry which is located in Bahadurgarh, Patiala district of Punjab, India. This industry is known to produce of ring carrier pistons, passenger car pistons and 4-stroke bi-wheeler pistons. The associated customers include the names of well reputed automobile firms such as Bajaj, TATA Motors, Mahindra, Escorts, Hero, Maruti Suzuki, Yamaha, Sonalika Internationals and Harley Davison Company. The main purpose of the industrial visit is to collect the field data. The field data refers to the data which is collected from the manufacturing system, during its working such as processing time, due dates, cycle time, and idle time, transportation distance, set up time and any break downs during the operation.

3.7.1 Process Flow-Piston Machinery Shop

Federal Mogul is a piston manufacturing industry, and manufacturing two different products, namely ring carrier pistons and 4-strokes bi-wheeler piston rings using a flow shop environment for the various machines. The flow shop arrangement authenticates the utilization of the same process sequence for all the products with an advantage of having different processing times on the various machines. Therefore, the flow patterns of both the products are common and are described with the help of flow chart in Figure 3.9.

3.7.2 Business Plan of Piston Manufacturing Industry

Business plan of a piston manufacturing industry enlists the essential factors according to which the manufacturing unit working is designed. These factors decide the number of products to be made in the unit per day, the rejection rate, cost of per product, and target under the specified period and so on. The various factors included in the business plan of the respective industry are listed below.

Business plan target for the month = 45,155

Production target for the month = 49,671

Actual production till date = 22,760

Asking rate of start of month = 1910 nos. /days

Asking rate for remaining WD's = 1416 nos. /days

Business plan rejection target = 4.20%

Actual rejection till date = 3.10%

Consumable cost/product (budgeted) = 7.50/production

Consumable cost/product (actual) = 6.15/production

Energy units/product (budgeted) = 0.38/production

Energy units/product (actual) = 0.37/production

3.7.3 Database Sheet

The preparation of the database sheet is an essential step in any industrial related research approach. These database sheets consists the information related to the respective manufacturing unit or shop. The sheet is prepared according to the need of data i.e. collection of the data needed for initializing the work. Some of the information in the sheets contains is enlisted below:

(a) Cycle time

(b) Processing time

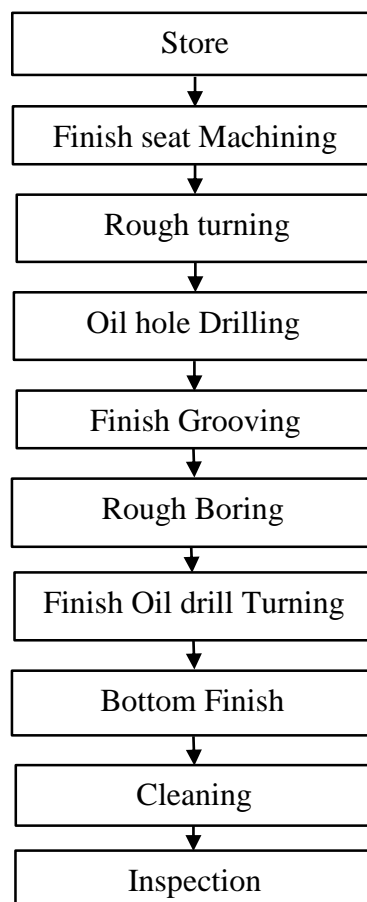


Figure 3.9: Depiction of the Flow Process of Products in Piston Manufacturing Industry

- (c) Due date
- (d) Number of workers
- (e) Work in process
- (f) Task Time

After deciding the activities to be worked on the process observation is conducted in which all the details about the process are collected. All the data needed is loaded in tables and graphs which is further used in formulation, analysis and documentation of the data. The observation should be very precise as the further activities depends on the observed data during the manufacturing in action.

3.7.4 Data Analysis and Documentation

After the collection of data, the drafting of the data is the next step to be executed which gives the current map of the various activities and times associated with them. The analysis of the data for each machine and part is necessary to ensure any non-existence of any errors in the collected data. The data may be compared with the previous obtained data for the same process and product The weak area which are identified to be worked on are closely analysed and respective actions are planned. Specific documentation is done for each process for better results.

3.7.4.1 Analysis of data of part 1 in the manufacturing industry

The analysis of data includes the different times taken on the various machines for the manufacturing of part 1. All the times are in minutes and the distance is in meters. The different notations used are: M1: Finish seat machining; M2: Rough turning; M3: Oil-hole

Table 3.2: Depiction of the Different Times of Part 1 on Various Machines

Part 1	M1	M2	M3	M4	M5	M6	M7	M8
Distance (m)	76	1.5	1.7	1.9	1.5	3	1.5	11.5
Transportation Time	5	0.5	0.5	0.5	0.5	0.5	0.5	3
Set Up Time	2	1.5	2	3	2	2	2.5	3.5
Process Time	11	11	12	19	11	15	19	15
Backlog time	4	5	5	8	4.5	6.5	11	10
Cycle time	17	18	19.5	30.5	18	24	33	31.5
Time/day	60	90	90	100	60	120	90	120

drilling; M4: Finish grooving; M5: Rough boring; M6: Finish diameter boring; M7: Bottom finish; M8: Cleaning; M9: Inspection; P1: Bi-wheeler piston rings; P2: Non-ring carrier. The following Table 3.2 shows the data analysis for Part 1:

3.7.4.2 Analysis of Data of Part 2 in the Manufacturing Industry

The analysis of the data, likewise for Part 1, here also describes the different times taken by the Part 2 on the different machines during its manufacturing. All times are in minutes and distance is in meters. The following Table 3.3 contains the analysis of data for part 2.

Table 3.3: Depiction of the Different Times of Part 2 on Various Machines

Part 2	M1	M2	M3	M4	M5	M6	M7	M8
Distance (m)	76	1.5	1.7	1.9	1.5	3	1.5	10
Transportation Time	10	0.5	0.5	0.5	0.5	0.5	0.5	1
Set Up Time	1.5	3	2	2.5	1	1.5	6	2
Process Time	12	11	15	21	13	14	20	17
Backlog time	5	3.5	6	10	5	6	8	10
Cycle time	18.5	18	23.5	34	19.5	22	34.5	30
Time/day	60	90	90	100	60	120	90	120

3.7.5 Formation of the Objective Functions for Part 1 and Part 2

The two objectives selected to formulate the data as a multi-objective problem are process time and the cycle time. The makespan and cycle time of both the products would be under consideration. Therefore, in order to minimize the process time and cycle time of the products the first and foremost step is to generate objective functions and subjected to constraints. Optimization is a process to produce the most beneficial, or optimum, solution for a root cause. An objective function can be minimized or maximized in the process. For our purpose, we need to minimize the value of the objective functions. The objective functions are made using the data analysis Table 3.2 and Table 3.3.

3.7.5.1 Objective functions for minimization of process time of part1 and part 2

For the optimization of make span, the following functions are to be minimized. The variable x (1) representative of Part 1 whereas x (2) represents Part 2. The coefficients of variables x (1) and x (2) represents the process time of Part 1 and Part2, respectively.

For Part 1 and Part 2 on machine 1:

$$11x(1) + 12x(2) \leq 60; \quad (3.1)$$

For Part 1 and Part 2 on machine 2:

$$11 x (1) + 11 x (2) \leq 90; \quad (3.2)$$

For Part 1 and Part 2 on machine 3:

$$12 x (1) + 15x (2) \leq 90; \quad (3.3)$$

For Part 1 and Part 2 on machine 4:

$$19 x (1) + 21 x (2) \leq 100; \quad (3.4)$$

For Part 1 and Part 2 on machine 5:

$$11 x (1) + 13 x (2) \leq 60; \quad (3.5)$$

For Part 1 and Part 2 on machine 6:

$$15 x (1) + 14 x (2) \leq 120; \quad (3.6)$$

For Part 1 and Part 2 on machine 7:

$$19 x (1) + 24 x (2) \leq 90; \quad (3.7)$$

For Part 1 and Part 2 on machine 8:

$$15 x (1) + 17 x (2) \leq 120; \quad (3.8)$$

3.7.5.2 Objective functions for minimization of cycle time of part 1 and part 2

For the optimization of cycle time, the following functions are to be minimized. The variable $x (1)$ is representative of Part 1 whereas $x (2)$ represents Part 2. The coefficients of variables $x (1)$ and $x (2)$ represents the cycle time of Part 1 and Part2, respectively.

For Part 1 and Part 2 on machine 1:

$$17 x (1) + 18.5 x (2) \leq 60; \quad (3.9)$$

For Part 1 and Part 2 on machine 2:

$$18 x (1) + 18 x (2) \leq 90; \quad (3.10)$$

For Part 1 and Part 2 on machine 3:

$$19.5 x (1) + 23.5 x (2) \leq 90; \quad (3.11)$$

For Part 1 and Part 2 on machine 4:

$$30.5 x (1) + 34 (x2) \leq 100; \quad (3.12)$$

For Part 1 and Part 2 on machine 5:

$$18 x (1) + 19.5 x (2) \leq 60; \quad (3.13)$$

For Part 1 and Part 2 on machine 6:

$$24 x (1) + 22 x (2) \leq 120; \quad (3.14)$$

For Part 1 and Part 2 on machine 7:

$$33 x (1) + 34.5 x (2) \leq 90; \quad (3.15)$$

For Part 1 and Part 2 on machine 8:

$$31.5 x (1) + 30 x(2) \leq 120; \quad (3.16)$$

3.8 DESCRIPTION OF SOFTWARE USED TO IMPLEMENT THE NOVEL HYBRID BBO-TLBO HEURISTIC FOR MFSP

The use of the some tools is always necessary to execute the loops designed in the algorithms. These software's repeat the steps in the loops with varying some of the values assigned by the user to obtain different results, each time. The variety in the results gives user the freedom to choose best results from the formed set of results. And, hence optimizes the values of the functions. As there are many software's available to code the algorithms some of which include C, C++, C#, Java, MATLAB, FORTRAN, LINGO with their different versions. The Software used to evaluate the functions of the subsections 3.7.1 and 3.7.2 is MATLAB 9.1 R2016b. The software was designed by Clever Moler in the 1970's and was first released in 1984. Nowadays this software is developed by 'Mathworks' while licensed by 'Proprietary Commercial Software'. Also known as Matrix laboratory, MATLAB is capable to handle the other languages such as C, C++, Java and Python as well. This software can solve inequality equations; object oriented programming, vectors, variables, structures such as arrays. Further, graphical user interfaces can be constructed which gives the user a chance to enter the values each time, also graphs in the three dimensions are possible to be designed and modifies, as required. In the present work, as there are sixteen different functions formed, these all will be framed separately in the software with names TB1, TB2... TB16. Each of the objective function will be evaluated using the developed BBO-TLBO heuristic, coded in the MATLAB as shown in the Section 3.6.3. The results obtained will be further compared to the actual collected data, and the optimized values will be computed. For loops is used in the programming of the algorithm to repeat the structure for the same function till 20 iterations. Arrays have been used to show the results in the sequential manner. The mathematical values obtained after solving the functions has been shown in Chapter 4 named as results and discussion. The present work in MATLAB uses quad-core processor with 4 GB RAM, system running on WINDOWS 10.

CHAPTER 4: RESULTS AND DISCUSSION

4.1 IMPLEMENTATION OF PROPOSED HYBRID BBO-TLBO HEURISTIC FOR MFSP

The proposed hybrid BBO-TLBO heuristic is implemented for the minimization of the two objectives, namely process time and cycle time which are sequence dependent setup time (SDST) correlating backlogging of all the machines for both the products. The proposed heuristic combines both BBO's migration and mutation phases with the teacher phase of TLBO. The migration phase includes immigration and emigration of the population and the mutation will preserve the diversity in the population. The teacher phase encourages the achieving the value near to the best solution. The heuristic will tend to minimize the function named as 'cost function value' as the number of iterations increases while using the different values of the variables each time for the successive iterations. The validation of the model is conducted by comparative analysis of actual results with computed values from MATLAB.

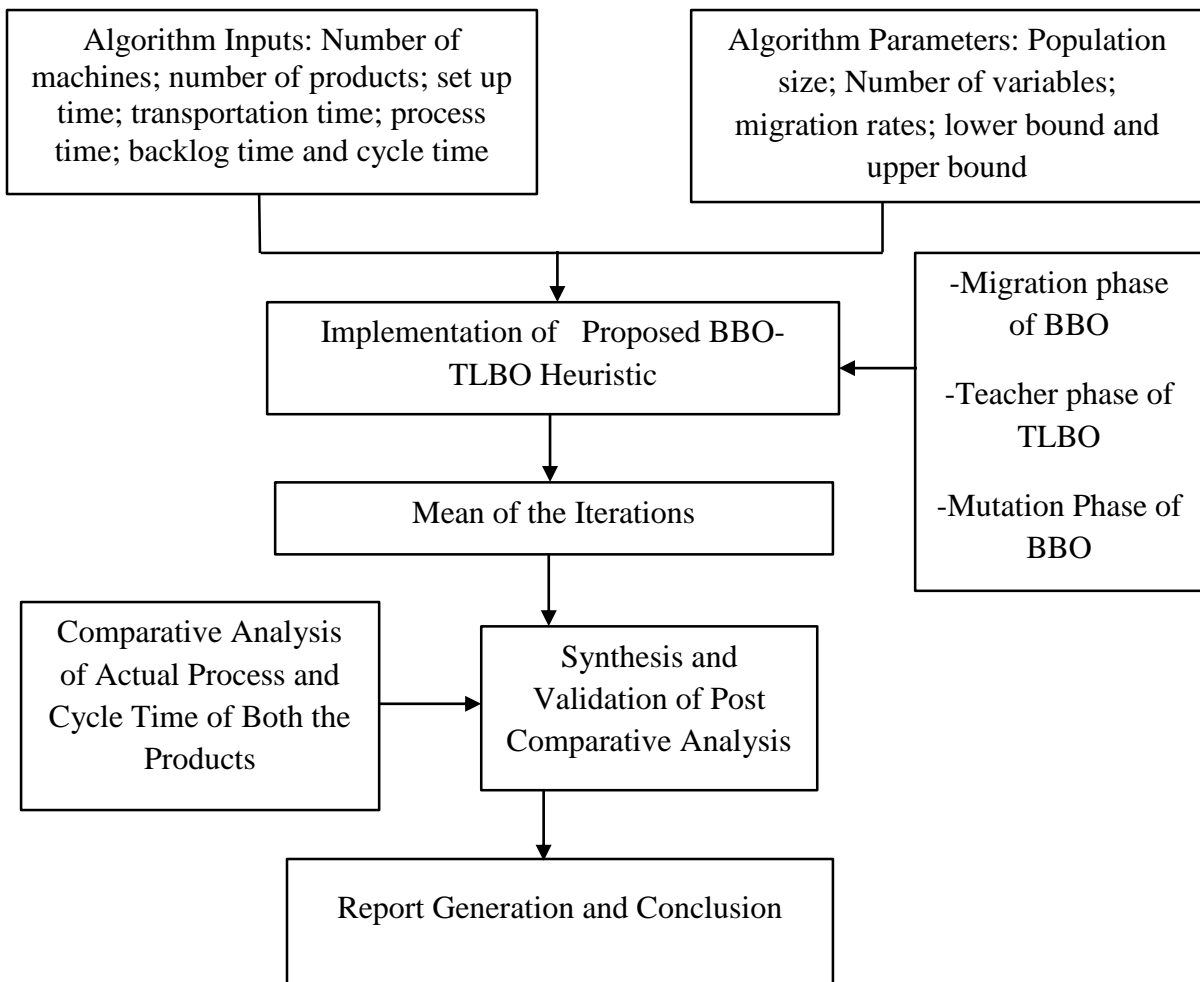


Figure 4.1: Implementation of Proposed BBO-TLBO Heuristic for MFSP

Further, the optimized values are also validated by the comparative analysis of functions of parameters and results generated from MATLAB.

4.2 MATHEMATICAL ANALYSIS USING PROPOSED BBO-TLBO FOR OPTIMIZATION OF PROCESS TIME AND CYCLE TIME FOR MFSP

The mathematical analysis includes the implementation of the proposed heuristic and analyzing the results produced for the respective functions. The heuristic will be producing the mathematical values using the algorithm explained in the Section 3.6.3. The objectives to be focused are minimization of process time and minimization of cycle time, respectively for each product.

Number of variables =2;

For functions related to minimization of process time,

Lower bound= 1, Upper Bound =15;

For functions related to minimization of cycle time,

Lower bound = 1, Upper Bound=24;

The upper bound is calculated by taking the average of the process time for the functions related to the minimization of process time, and taking average of cycle time for the functions related to minimization of the cycle time, for both the products.

Population Size=20,

Number of Iterations =20,

Mutation rate=0.1,

Keep Rate =0.2,

Equation of teacher phase,

$X_{\text{new}} = X_{\text{old}} + (X_{\text{teacher}} - (T_F) \text{ Mean})$, where $X_{\text{new}}, X_{\text{old}}$ = new solution and old solution respectively Mean= average of the population, T_F = Teaching factor ranging between 1 and 2.

4.2.1 Mathematical Analysis for Minimization of Process Time Using the Proposed BBO-TLBO Heuristic

This section is addressing the evaluation and optimization of various functions of process time formulated in Section 3.7.5.1 with the proposed hybrid BBO-TLBO heuristic algorithm. These respective iterations of functions with analytical results are evaluated and explained below as:

(i) Function 1: Evaluation and Optimization of function $11x(1)+12x(2)\leq 60$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.1: Evaluation and Optimization of Function $11x(1)+12 x(2)\leq 60$ Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost Function Value
Iteration 1	10.700	10.868	248.132
Iteration 2	6.899	13.741	240.868
Iteration 3	4.887	14.865	232.199
Iteration 4	4.391	13.796	213.826
Iteration 5	4.639	13.142	208.726
Iteration 6	10.638	6.962	200.514
Iteration 7	4.568	6.366	192.314
Iteration 8	10.447	4.904	191.313
Iteration 9	10.964	7.593	179.465
Iteration 10	7.697	7.532	175.056
Iteration 11	7.410	7.172	167.586
Iteration 12	9.059	4.132	149.253
Iteration 13	5.144	6.461	134.1209
Iteration 14	1.849	8.364	120.718
Iteration 15	9.092	1.249	115.005
Iteration 16	2.087	7.123	108.445
Iteration 17	7.592	1.065	96.298
Iteration 18	1.552	5.493	82.991
Iteration 19	4.934	1.349	70.465
Iteration 20	2.748	1.971	53.891
Mean	6.365	7.402	

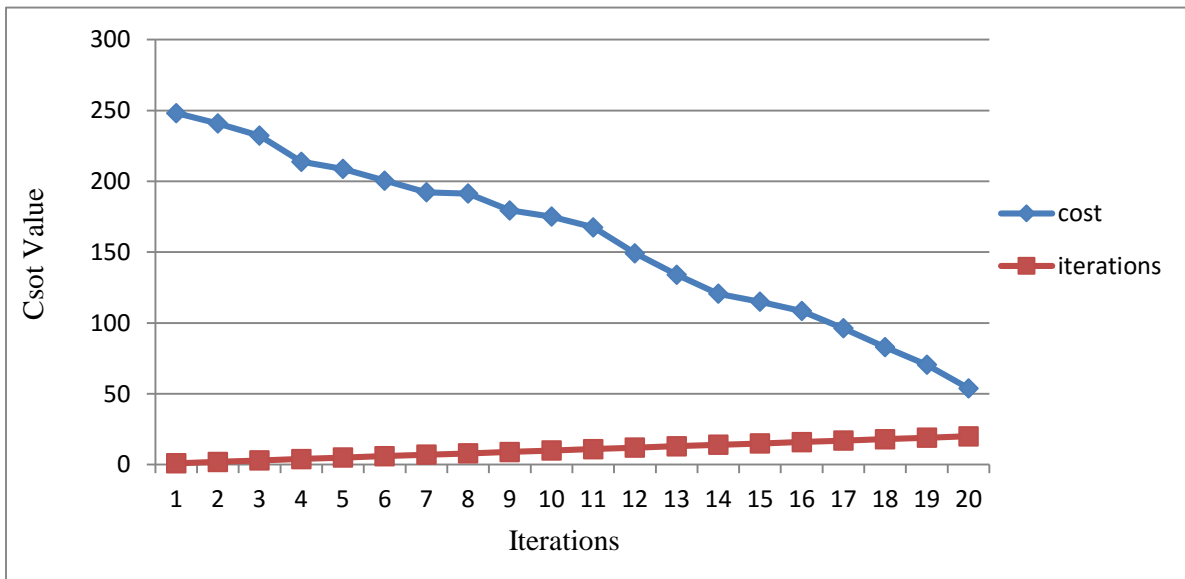


Figure 4.2: Graph Showing the Variation of Function $f=11x(1)+12 x(2)\leq 60$ Corresponding to the Number of Iterations

Table 4.1 addresses the evaluation and optimization of the function $f=11x(1)+12x(2)\leq 60$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 15 which further impacts the optimization of the evaluated Cost function value. The variable $x(1)$ = processing time of part 1 on machine 1 and $x(2)$ = processing time of part 2 on machine 1. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<60$. The mean of iterations for $x(1)$ comes out to be 6.365 and that of $x(2)$ is 7.402. The best cost function value is obtained is 53.891. Figure 4.2 illustrates the variation of the values of the function $f=11x(1)+12x(2)\leq 60$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 248.132 is at first iteration and there after optimized to 50.891 at the last. The minimum value of the function can be seen on the 20th iteration. The dots in the graph represents, the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

Table 4.2: Evaluation and Optimization of Function Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost Function Value
Iteration 1	12.872	13.173	286.506
Iteration 2	10.939	12.803	261.175
Iteration 3	11.880	11.113	252.927
Iteration 4	13.334	8.787	243.345
Iteration 5	8.341	13.327	238.354
Iteration 6	13.268	4.489	195.342
Iteration 7	5.905	11.80	194.818
Iteration 8	14.798	2.85	194.224
Iteration 9	4.477	13.176	194.193
Iteration 10	3.614	11.462	165.845
Iteration 11	8.724	5.260	153.828
Iteration 12	6.619	6.495	144.264
Iteration 13	6.414	6.224	139.028
Iteration 14	10.546	2.053	138.596
Iteration 15	9.371	3.186	138.138
Iteration 16	9.772	2.651	136.661
Iteration 17	4.464	6.319	118.632
Iteration 18	5.745	4.678	114.672
Iteration 19	2.144	3.723	64.546
Iteration 20	1.817	1.741	39.147
Mean	8.252	7.266	

(ii) Function 2: Evaluation and Optimization of function $11x(1)+11x(2)\leq 90$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.2 addresses the evaluation and optimization of the function $f=11x(1)+11x(2)\leq 90$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 15 which further impacts the optimization of the evaluated Cost function value. The variable $x(1)$ = processing time of part 1 on machine 2 and $x(2)$ = processing time of part 2 on machine 2. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<90$. The mean of iterations for $x(1)$ comes out to be 8.252 and that of $x(2)$ is 7.266. The best cost function value is obtained is 39.147. Figure 4.3 refers the variation of the values of the function $f=11x(1)+11x(2)\leq 90$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 286.506 is at first iteration and there after optimized to 39.147 at the last. The minimum value of the function can be seen on the 20th iteration. The dots in the graph represents, the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

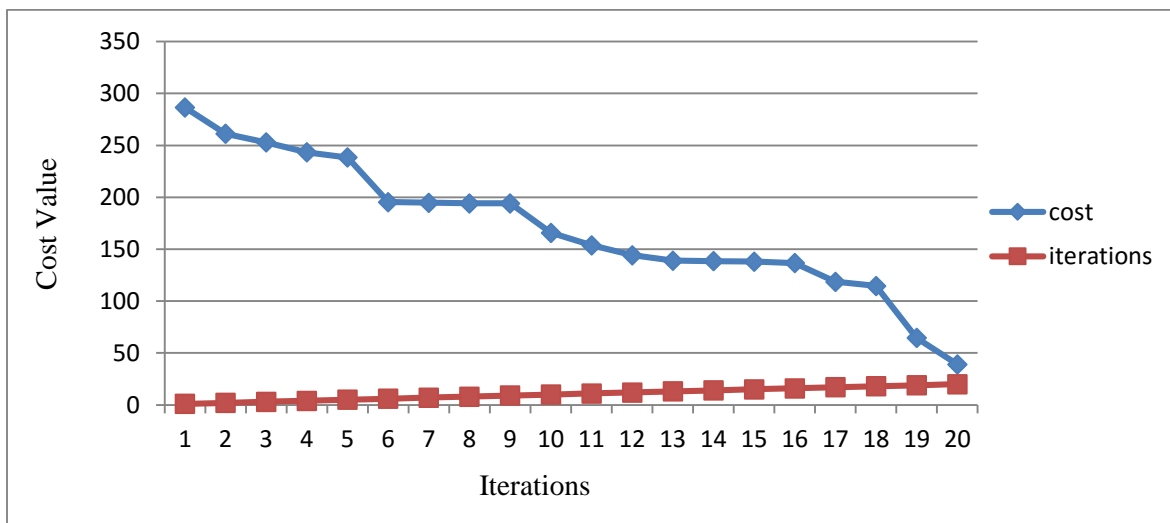


Figure 4.3: Graph Showing the Variation of Function $f=11x(1)+11x(2)\leq 90$ Corresponding to the Number of Iterations

(iii) Function 3: Evaluation and Optimization of function $12x(1)+15x(2)\leq 90$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.3 addresses the evaluation and optimization of the function $f =12x(1)+15x(2)\leq 90$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the

range between 1 and 15 which further impacts the optimization of the evaluated cost function value.

Table 4.3: Evaluation and Optimization of Function $12x(1)+15x(2)\leq 90$ Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost Function Value
Iteration 1	13.964	12.793	359.482
Iteration 2	8.907	14.621	326.21
Iteration 3	8.660	13.870	311.984
Iteration 4	7.252	14.35	302.416
Iteration 5	14.099	7.461	281.109
Iteration 6	6.023	13.44	273.927
Iteration 7	11.259	7.943	254.265
Iteration 8	3.695	13.50	246.956
Iteration 9	11.937	6.850	246.008
Iteration 10	12.245	5.189	225.652
Iteration 11	10.615	6.303	221.933
Iteration 12	10.603	5.599	211.237
Iteration 13	14.384	2.309	207.253
Iteration 14	13.68	1.543	187.324
Iteration 15	9.409	4.059	173.803
Iteration 16	4.393	7.396	159.061
Iteration 17	3.422	6.668	141.086
Iteration 18	2.927	4.582	103.867
Iteration 19	6.968	1.005	98.703
Iteration 20	1.889	1.627	47.082
Mean	8.816	7.565	

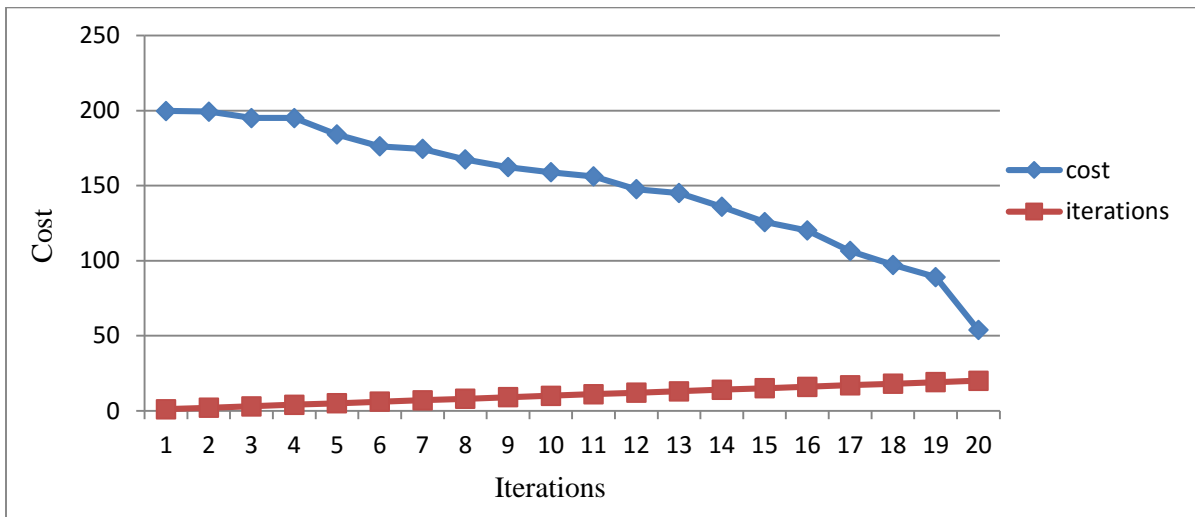


Figure 4.4: Graph Showing the Variation of Function $f = 12 x (1) + 15 x (2) \leq 90$ Corresponding to the Number of Iterations

The variable $x(1)$ =processing time of part 1 on machine 3 and $x(2)$ = processing time of part 2 on machine 3. The constraints in the equation are $x(1) > 0$, $x(2) > 0$ and $f < 90$. The mean of iterations for $x(1)$ comes out to be 8.816 and that of $x(2)$ is 7.565. The best cost function value is obtained is 47.082. Figure 4.4 depicts the variation of values of the function $f = 12x(1) + 15x(2) \leq 90$ with the iterations taking place. It is evident from the graph that as number of iterations are increasing, the value of the function keeps optimizing. The maximum value 359.482 is at first iteration and there after optimized to 47.082 at the last. The minimum value of the function can be seen on the 20th iteration. The dots represents the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

(iv) Function 4: Evaluation and Optimization of function $19x(1)+21x(2)\leq 100$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.4: Evaluation and Optimization of Function $19x(1)+21x(2)\leq 100$ Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost Function Value
Iteration 1	13.019	12.416	508.111
Iteration 2	10.288	13.288	474.543
Iteration 3	14.707	7.177	430.175
Iteration 4	8.201	13.042	429.716
Iteration 5	8.083	10.469	373.440
Iteration 6	5.063	12.762	364.221
Iteration 7	8.364	9.729	363.259
Iteration 8	6.981	10.219	347.250
Iteration 9	5.522	10.11	308.148
Iteration 10	5.760	9.462	317.405
Iteration 11	5.840	9.285	305.946
Iteration 12	11.495	3.971	301.821
Iteration 13	14.046	1.325	294.724
Iteration 14	10.855	3.273	275.004
Iteration 15	12.238	1.519	264.429
Iteration 16	6.439	6.649	261.990
Iteration 17	7.736	4.773	247.236
Iteration 18	6.469	4.129	209.629
Iteration 19	5.519	4.944	208.709
Iteration 20	2.577	2.720	106.110
Mean	8.460	7.564	

Table 4.4 addresses the evaluation and optimization of the function $f=19x(1)+21x(2)\leq 100$, by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 15 which further impacts the optimization of the evaluated Cost function value. The variable $x(1)$ = processing time of part 1 on machine 4 and $x(2)$ = processing time of part 2 on machine 4. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<60$. The mean of iterations for $x(1)$ comes out to be 8.460 and that of $x(2)$ is 7.564. Figure 4.5 illustrates the variation of the values of the function $f=19x(1)+21x(2)\leq 100$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 508.111 is at first iteration and there after optimized to 106.110 at the last. The minimum value of the function can be seen on the 20th iteration. The dots in the graph represents, the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

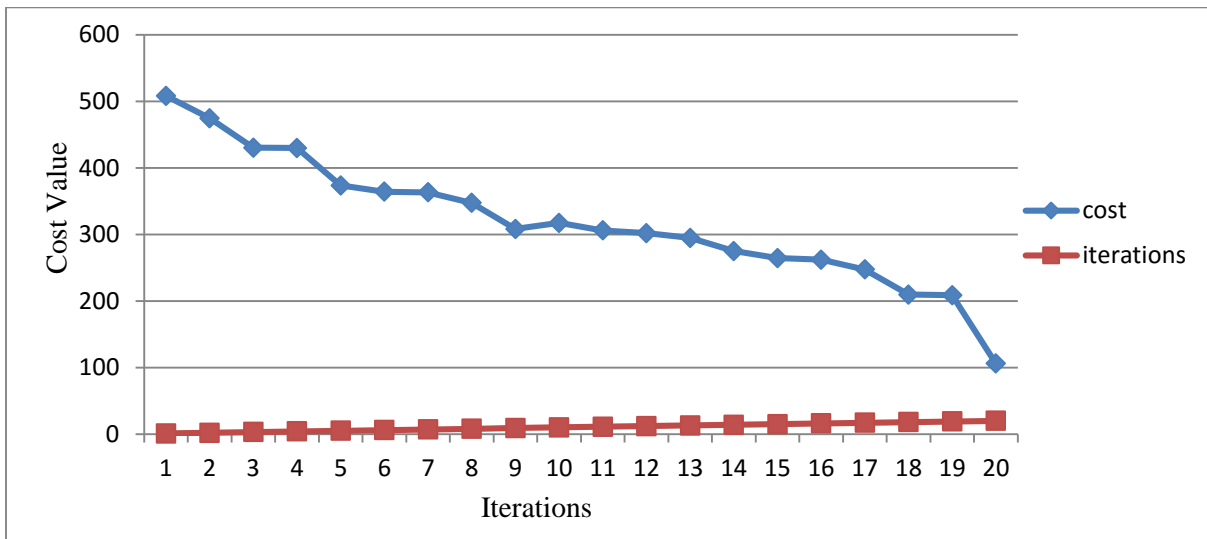


Figure 4.5: Graph Showing the Variation of Function $f=19x(1)+21x(2)\leq 100$ Corresponding to the Number of Iterations

(v) Function 5: Evaluation and Optimization of function $11x(1) + 13x(2) \leq 60$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.5 addresses the evaluation and optimization of the function $f=11x(1)+13x(2)\leq 60$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 15 which further impacts the optimization of the evaluated cost function

value. The variable $x(1)$ = processing time of part 1 on machine 5 and $x(2)$ = processing time of part 2 on machine 5. The constraints in the equation are $x(1) > 0$, $x(2) > 0$ and $f < 60$.

Table 4.5: Evaluation and Optimization of Function $11x(1)+13x(2) \leq 60$ Using Proposed BBO- TLBO Heuristic

Iterations	$x(1)$	$x(2)$	Cost function value
Iteration 1	14.405	14.424	345.971
Iteration 2	12.887	14.075	324.752
Iteration 3	12.090	14.432	320.627
Iteration 4	12.406	13.681	314.321
Iteration 5	10.502	11.608	266.433
Iteration 6	14.400	7.795	259.742
Iteration 7	6.905	13.820	255.615
Iteration 8	3.206	14.588	224.920
Iteration 9	11.403	6.4911	209.827
Iteration 10	2.777	13.787	209.790
Iteration 11	2.359	12.528	188.827
Iteration 12	12.203	5.439	188.715
Iteration 13	10.727	1.482	176.603
Iteration 14	14.303	2.986	173.066
Iteration 15	4.898	8.656	166.421
Iteration 16	10.176	3.396	156.099
Iteration 17	9.853	2.36	139.135
Iteration 18	10.884	1.479	138.524
Iteration 19	10.180	1.499	131.483
Iteration 20	4.876	1.641	75.049
Mean	9.572	8.311	

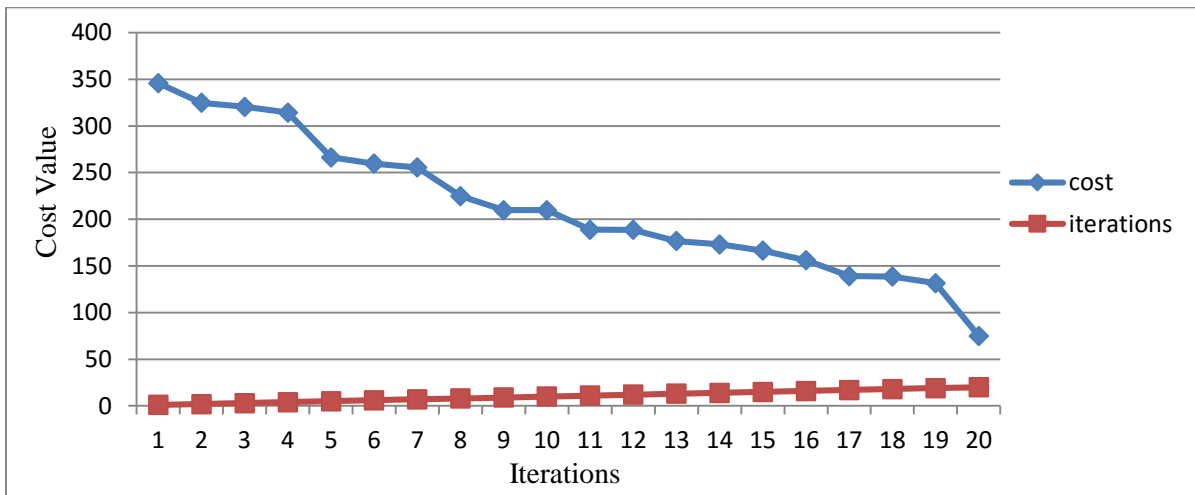


Figure 4.6: Graph Showing the Variation of Function $f=11x(1)+13x(2) \leq 60$ Corresponding to the Number of Iterations

The mean of iterations for $x(1)$ comes out to be 9.572 and that of $x(2)$ is 8.311. The best Cost function value is obtained is 75.049. Figure 4.6 depicts the variation of the values of the function $f=11x(1)+13x(2)\leq 60$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 345.971 is at first iteration and there after optimized to 75.049 at the last. The minimum value of function can be seen on 20th iteration. The dots in the graph represents, the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

(vi) Function 6: Evaluation and Optimization of function $15x(1)+14x(2)\leq 120$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.6: Evaluation and Optimization of Function $15x(1)+14x(2)\leq 120$ Using Proposed BBO- TLBO Heuristic

Iterations	$x(1)$	$x(2)$	Cost function value
Iteration 1	13.729	13.080	389.070
Iteration 2	12.340	14.556	388.897
Iteration 3	13.756	11.047	361.022
Iteration 4	9.939	14.901	357.717
Iteration 5	9.515	14.409	342.909
Iteration 6	14.615	8.560	339.083
Iteration 7	9.412	13.976	338.402
Iteration 8	10.133	10.567	299.951
Iteration 9	11.082	8.184	280.817
Iteration 10	6.980	12.112	274.289
Iteration 11	11.323	5.095	241.198
Iteration 12	8.592	7.477	233.576
Iteration 13	12.261	1.111	199.455
Iteration 14	1.960	11.635	192.301
Iteration 15	1.585	9.864	161.882
Iteration 16	1.604	9.637	159.010
Iteration 17	8.693	1.101	145.824
Iteration 18	7.888	1.356	137.312
Iteration 19	7.756	1.421	133.713
Iteration 20	1.288	1.802	44.559
Mean	8.723	8.592	

Table 4.6 addresses the evaluation and optimization of the function $f=15x(1)+14x(2)\leq 120$, by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follow the range between 1 and 15 which further impacts the optimization of the evaluated Cost function

value. The variable $x(1)$ =processing time of part 1 on machine 6 and $x(2)$ = processing time of part 2 on machine 6. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<120$. The mean of iterations for $x(1)$ comes out to be 8.723 and that of $x(2)$ is 8.592. The best Cost function value is obtained is 44.559. Figure 4.7 illustrates the variation of the values of the function $f=15x(1)+14x(2)\leq 120$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 389.070 is at first iteration and there after optimized to 44.559 at the last. The minimum value of the function can be seen on the 20th iteration. The dots in the graph represents, the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

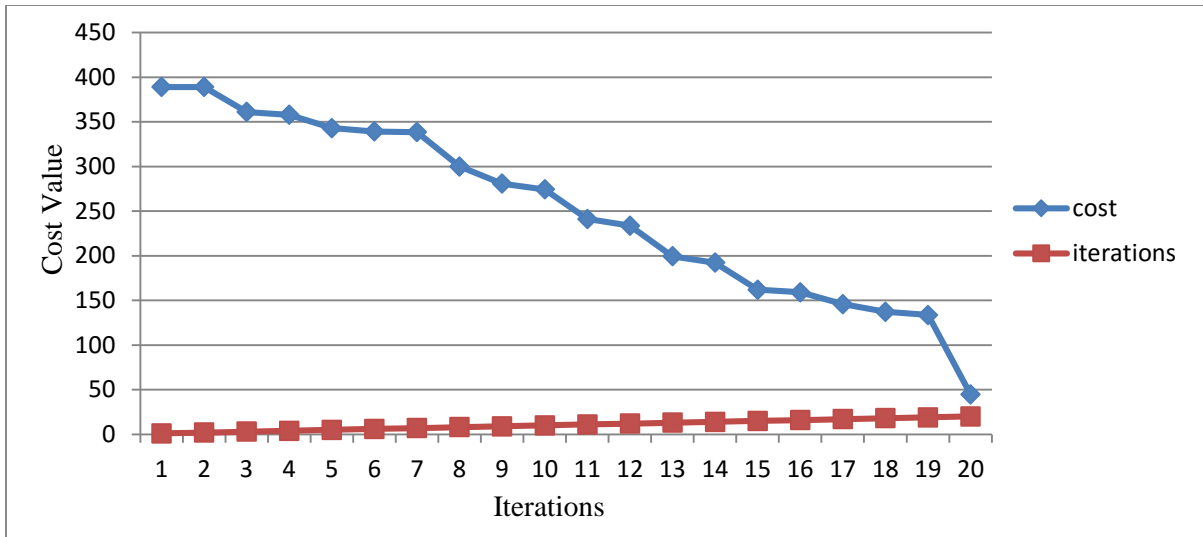


Figure 4.7: Graph Showing the Variation of Function $f=15x(1)+14x(2)\leq 120$ Corresponding to the Number of Iterations

(vii) Function 7: Evaluation and Optimization of function $19x(1)+24x(2)\leq 90$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.7 addresses the evaluation and optimization of the function $f=19x(1)+24x(2)\leq 90$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 15 which further impacts the optimization of the evaluated cost function value. The variable $x(1)$ =processing time of part 1 on machine 7 and $x(2)$ =processing time of part 2 on machine 7. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<90$. The mean of iterations for $x(1)$ comes out to be 7.613 and that of $x(2)$ is 8.343. The best Cost function value is obtained is 178.588.

Table 4.7: Evaluation and Optimization of Function $f=19x(1)+24x(2)\leq 90$ Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost Function Value
Iteration 1	14.924	11.354	556.059
Iteration 2	13.394	10.817	514.12
Iteration 3	9.963	13.120	504.195
Iteration 4	8.003	12.652	455.722
Iteration 5	10.921	9.957	446.497
Iteration 6	12.056	7.773	415.631
Iteration 7	5.340	12.773	408.051
Iteration 8	3.655	13.463	392.584
Iteration 9	9.996	7.931	380.288
Iteration 10	4.856	10.926	354.524
Iteration 11	6.125	9.027	336.028
Iteration 12	5.275	8.837	312.347
Iteration 13	7.277	6.542	295.303
Iteration 14	6.996	6.292	283.958
Iteration 15	5.602	5.602	240.916
Iteration 16	5.918	4.942	231.070
Iteration 17	7.008	3.242	210.994
Iteration 18	1.371	7.584	208.152
Iteration 19	7.151	1.897	181.410
Iteration 20	6.426	2.359	178.588
Mean	7.613	8.343	

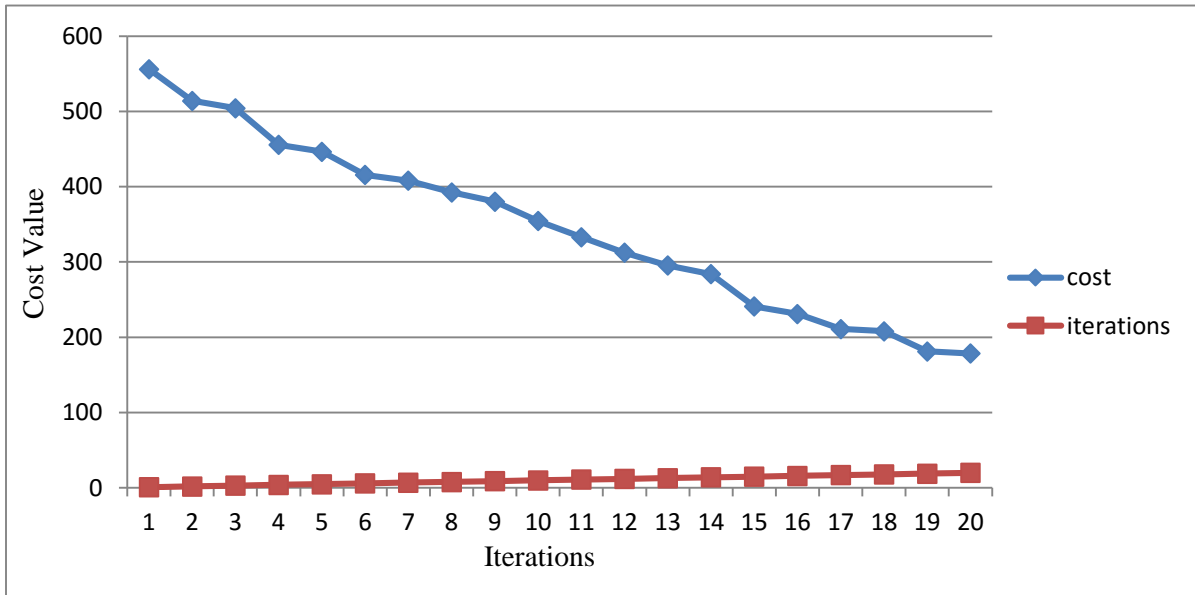


Figure 4.8: Graph Showing the Variation of Function $f=19x(1)+24x(2)\leq 90$ Corresponding to the Number of Iterations

Figure 4.8 refers the variation of the values of the function $f=19x(1)+24x(2)\leq 90$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 556.059 is at first iteration and there after optimized to 178.588 at the last. The minimum value of the function can be seen on the 20th iteration. The dots in the graph represents, the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

(viii) Function 8: Evaluation and Optimization of function $15x(1) + 17x(2) \leq 120$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.8: Evaluation and Optimization of Function $15x(1)+17x(2)\leq 120$ Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost function value
Iteration 1	14.258	14.692	463.758
Iteration 2	11.563	12.069	378.637
Iteration 3	8.105	13.903	357.941
Iteration 4	12.050	9.091	335.321
Iteration 5	13.444	7.085	322.158
Iteration 6	14.762	5.615	316.903
Iteration 7	7.323	11.488	305.151
Iteration 8	8.006	10.760	303.039
Iteration 9	6.857	11.364	296.067
Iteration 10	4.282	12.672	279.068
Iteration 11	1.717	14.645	274.692
Iteration 12	9.820	7.179	269.359
Iteration 13	10.838	6.142	266.992
Iteration 14	13.761	1.578	233.258
Iteration 15	12.861	1.992	226.781
Iteration 16	11.703	2.983	226.265
Iteration 17	5.076	8.671	223.559
Iteration 18	7.197	6.530	218.991
Iteration 19	11.959	1.861	211.029
Iteration 20	1.285	4.352	87.518
Mean	9.343	8.206	

Table 4.8 addresses the evaluation and optimization of the function $f=15x(1)+17x(2)\leq 120$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 15 which further impacts the optimization of the evaluated cost function value. The variable $x(1)$ =processing time of part 1 on machine 8 and $x(2)$ =processing time of part 2 on machine 8. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<120$. The mean of iterations for $x(1)$ comes out to be 9.343 and that of $x(2)$ is 8.206. The best Cost function

value is obtained is 87.518. Figure 4.9 displays the variation of the values of the function $f=15x(1)+17x(2)\leq 120$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 463.758 is at first iteration and there after optimized to 87.518 at the last. The minimum value of the function can be seen on the 20th iteration.

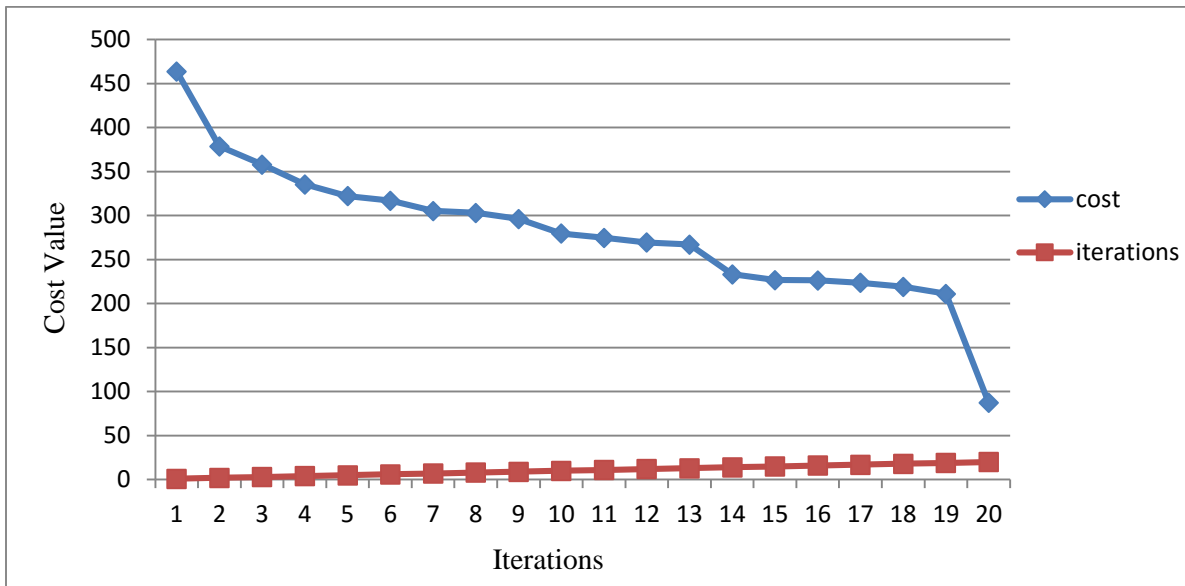


Figure 4.9: Graph Showing the Variation of Function $f = 15x(1)+17x(2)\leq 120$ Corresponding to the Number of Iterations

4.2.2 Mathematical Analysis for Minimization of Cycle Time Using the Proposed Hybrid BBO-TLBO Heuristic

This section is addressing the evaluation and optimization of various functions of cycle time which includes sequence dependent setup time (SDST) and backlogging time formulated in Section 3.7.5.2 with the proposed hybrid BBO-TLBO heuristic algorithm. These respective iterations of functions with analytical results are evaluated and explained below as:

(i) Function 9: Evaluation and Optimization of function $17x(1)+18.5x(2)\leq 60$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.9 addresses the evaluation and optimization of the function $f=17x(1)+18.5x(2)\leq 60$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 24 which further impacts the optimization of the evaluated cost function value. The variable $x(1)$ =processing time of part 1 on machine 1 and $x(2)$ = processing time of part 2 on machine 1. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<60$. The mean of

iterations for $x(1)$ comes out to be 13.139 and that of $x(2)$ is 12.106. The best cost function value is obtained is 235.570.

Table 4.9: Evaluation and Optimization of Function $f=17x(1)+18.5x(2)\leq 60$ Using Proposed BBO- TLBO Heuristic

Iterations	$x(1)$	$x(2)$	Cost Function Value
Iteration 1	23.770	20.075	775.506
Iteration 2	22.901	14.604	659.506
Iteration 3	9.987	18.802	517.640
Iteration 4	4.365	23.640	511.558
Iteration 5	21.971	6.483	493.565
Iteration 6	21.033	7.312	492.850
Iteration 7	12.758	14.790	490.528
Iteration 8	18.611	9.224	487.057
Iteration 9	7.005	19.130	473.018
Iteration 10	20.602	6.135	463.744
Iteration 11	3.395	21.858	462.095
Iteration 12	20.446	4.563	432.006
Iteration 13	12.514	11.690	429.006
Iteration 14	11.516	11.359	405.925
Iteration 15	9.196	13.420	404.625
Iteration 16	10.167	9.420	347.128
Iteration 17	15.437	2.557	309.735
Iteration 18	5.966	10.406	293.953
Iteration 19	8.464	6.507	264.291
Iteration 20	2.680	10.270	235.570
Mean	13.139	12.106	

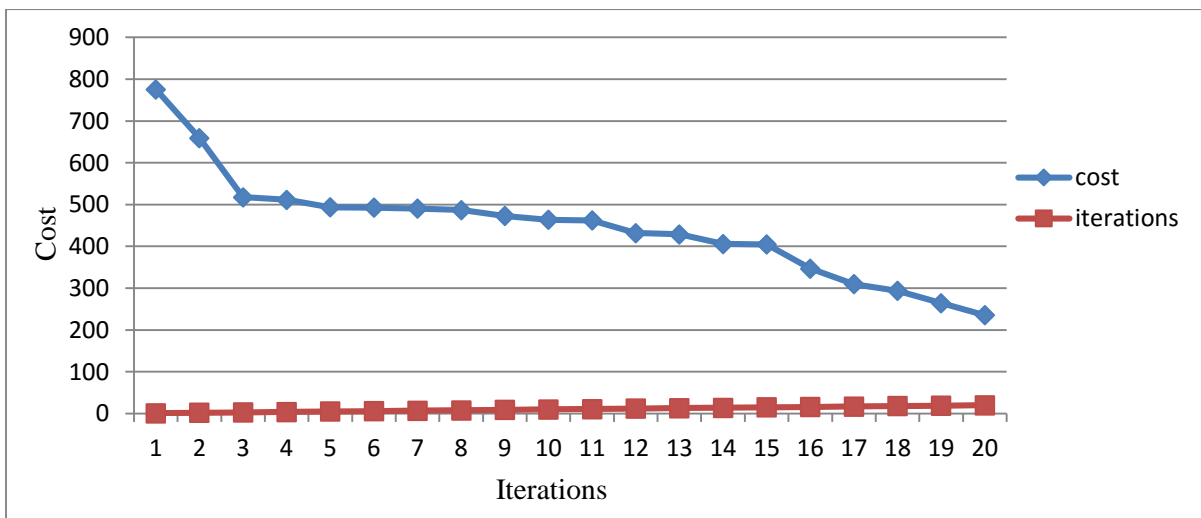


Figure 4.10: Graph Showing the Variation of Function $f=17x(1)+18.5x(2)\leq 60$ Corresponding to the Number of Iterations

Figure 4.10 illustrates the variation of the values of the function $f=17x(1)+18.5x(2)\leq 60$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 775.506 is at first iteration and there after optimized to 235.570 at the last. The dots in the graph represents, the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

(ii) Function 10: Evaluation and Optimization of function $18x(1)+18x(2)\leq 90$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.10: Evaluation and Optimization of Function $18x(1)+18x(2)\leq 90$ Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost Function Value
Iteration 1	22.062	17.749	716.629
Iteration 2	18.425	18.676	667.840
Iteration 3	20.390	13.779	615.072
Iteration 4	11.567	21.262	590.933
Iteration 5	23.748	7.974	571.011
Iteration 6	13.561	17.090	551.741
Iteration 7	11.053	17.145	507.579
Iteration 8	15.407	11.433	483.133
Iteration 9	6.562	19.411	467.525
Iteration 10	16.356	5.585	395.001
Iteration 11	6.459	13.401	357.509
Iteration 12	8.626	9.943	334.255
Iteration 13	11.518	7.009	333.498
Iteration 14	13.241	4.732	323.538
Iteration 15	4.162	13.750	322.431
Iteration 16	11.129	5.312	295.95
Iteration 17	2.266	12.939	273.718
Iteration 18	1.346	12.652	251.979
Iteration 19	8.727	4.635	247.319
Iteration 20	1.904	9.555	206.277
Mean	11.426	12.169	

Table 4.10 addresses the evaluation and optimization of the function $18x(1)+18x(2)\leq 90$ by the proposed BBO-TLBO heuristic. In this table, the input variables x(1) and x (2) follows the range between 1 and 24 which further impacts the optimization of the evaluated cost function value. The variable x(1)=processing time of part 1 on machine 2 and x(2)= processing time of part 2 on machine 2. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<90$. The mean of iterations for x (1) comes out to be 11.426 and that of x(2) is 12.169. The best cost function value is obtained is 247.319. Figure 4.11 depicts the variation of the values of the function

$18x(1)+18x(2)\leq 90$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 716.629 is at first iteration and there after optimized to 247.319 at the last. The minimum value of the function can be seen on the last iteration.

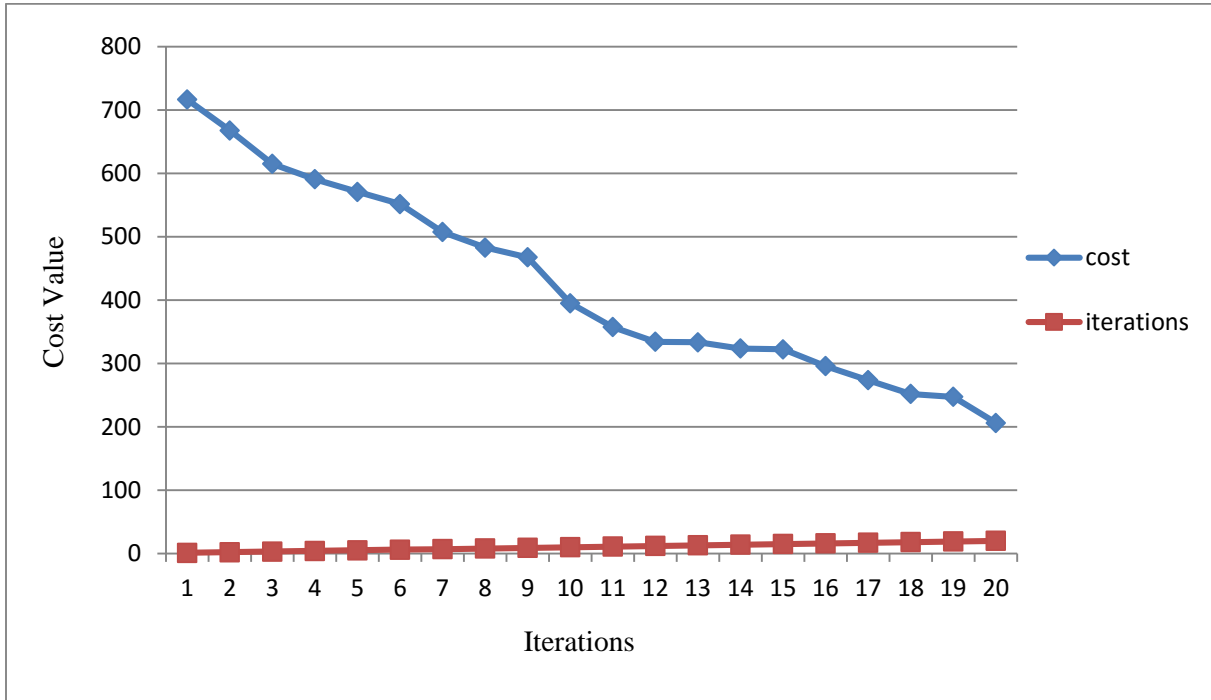


Figure 4.11: Graph Showing the Variation of Function $f=18x(1)+18x(2)\leq 90$ Corresponding to the Number of Iterations

(iii) Function 11: Evaluation and Optimization of function $19.5x(1)+23.5x(2)\leq 90$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.11 addresses the evaluation and optimization of the function $f=19.5x(1)+23.5x(2)\leq 90$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 24 which further impacts the optimization of the evaluated cost function value. The variable $x(1)$ =processing time of part 1 on machine 3 and $x(2)$ = processing time of part 2 on machine 3. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<90$. The mean of iterations for $x(1)$ comes out to be 13.013 and that of $x(2)$ is 12.955. The best cost function value is obtained is 110.574. Figure 4.12 illustrates the variation of the values of the function $f=19.5x(1)+23.5x(2)\leq 90$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 908.782 is at first iteration and there after optimized to 110.574 at the last.

Table 4.11: Evaluation and Optimization of Function $19.5x(1)+23.5x(2)\leq 90$ Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost Function Value
Iteration 1	19.426	22.551	908.785
Iteration 2	23.894	18.028	889.625
Iteration 3	15.951	21.423	814.488
Iteration 4	17.835	18.850	790.794
Iteration 5	14.912	20.743	778.274
Iteration 6	9.945	23.277	740.961
Iteration 7	11.087	19.903	683.920
Iteration 8	16.241	15.361	677.710
Iteration 9	21.692	9.189	638.977
Iteration 10	9.627	18.976	633.680
Iteration 11	7.289	20.430	622.027
Iteration 12	17.049	6.979	496.488
Iteration 13	14.557	7.432	458.533
Iteration 14	21.488	1.174	446.640
Iteration 15	8.823	7.518	348.748
Iteration 16	6.361	9.460	346.371
Iteration 17	5.937	9.402	336.734
Iteration 18	10.186	2.395	254.943
Iteration 19	6.391	2.774	189.794
Iteration 20	1.558	3.412	110.574
Mean	13.013	12.955	

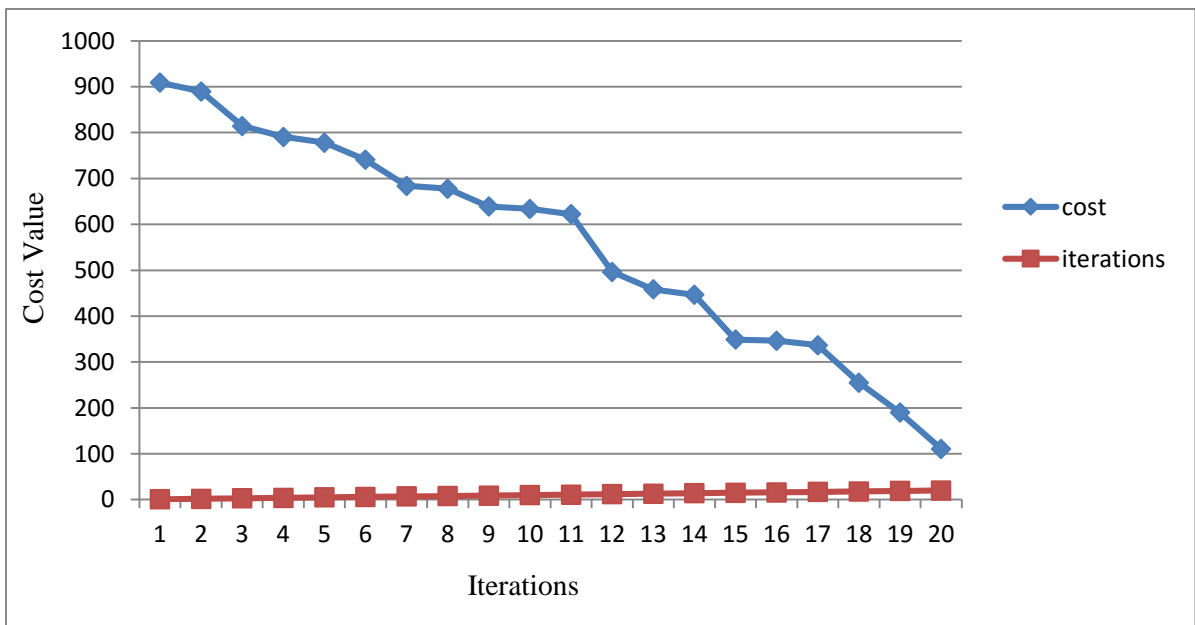


Figure 4.12: Graph Showing the Variation of Function $f = 19.5x(1)+23.5x(2)\leq 90$ Corresponding to the Number of Iterations

The minimum value of the function can be seen on the 20th iteration. The dots in the graph represents, the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

(iv) Function 12: Evaluation and Optimization of function $30.5x(1)+34x(2)\leq 100$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.12 addresses the evaluation and optimization of the function $f=30.5x(1)+34x(2)\leq 100$ by the proposed BBO-TLBO heuristic. In this table, input variables $x(1)$ and $x(2)$ follows range between 1 and 24 which further impacts the optimization of the evaluated Cost function value. The variable $x(1)$ =processing time of part 1 on machine 4 and $x(2)$ = processing time of part 2 on machine 4. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<100$.

Table 4.12: Evaluation and Optimization of Function $30.5x(1)+34x(2)\leq 100$ Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost Function Value
Iteration 1	23.026	23.054	1486.031
Iteration 2	20.529	22.481	1390.547
Iteration 3	19.220	23.068	1370.557
Iteration 4	19.738	21.833	1344.358
Iteration 5	16.610	18.428	1133.185
Iteration 6	23.018	12.163	1115.512
Iteration 7	10.700	22.061	1076.471
Iteration 8	4.625	23.323	934.061
Iteration 9	18.092	10.021	892.529
Iteration 10	3.920	22.007	868.841
Iteration 11	16.981	8.293	799.894
Iteration 12	3.234	19.939	776.582
Iteration 13	22.855	1.792	758.017
Iteration 14	19.406	4.263	736.851
Iteration 15	7.405	13.578	687.527
Iteration 16	16.075	4.937	658.185
Iteration 17	17.239	1.732	584.684
Iteration 18	15.544	3.298	584.376
Iteration 19	16.082	1.821	552.428
Iteration 20	7.369	2.061	294.867
Mean	15.083	13.012	

The mean of iterations for $x(1)$ comes out to be 15.083 and that of $x(2)$ is 13.012. The best cost function value is obtained is 294.867. Figure 4.13 illustrates the variation of the values of the function $f=30.5x(1)+34x(2)\leq 100$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing.

The maximum value 1486.031 is at first iteration and there after optimized to 294.867 at the last. The minimum value of the function can be seen on the 20th iteration. The dots in the graph represents, the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

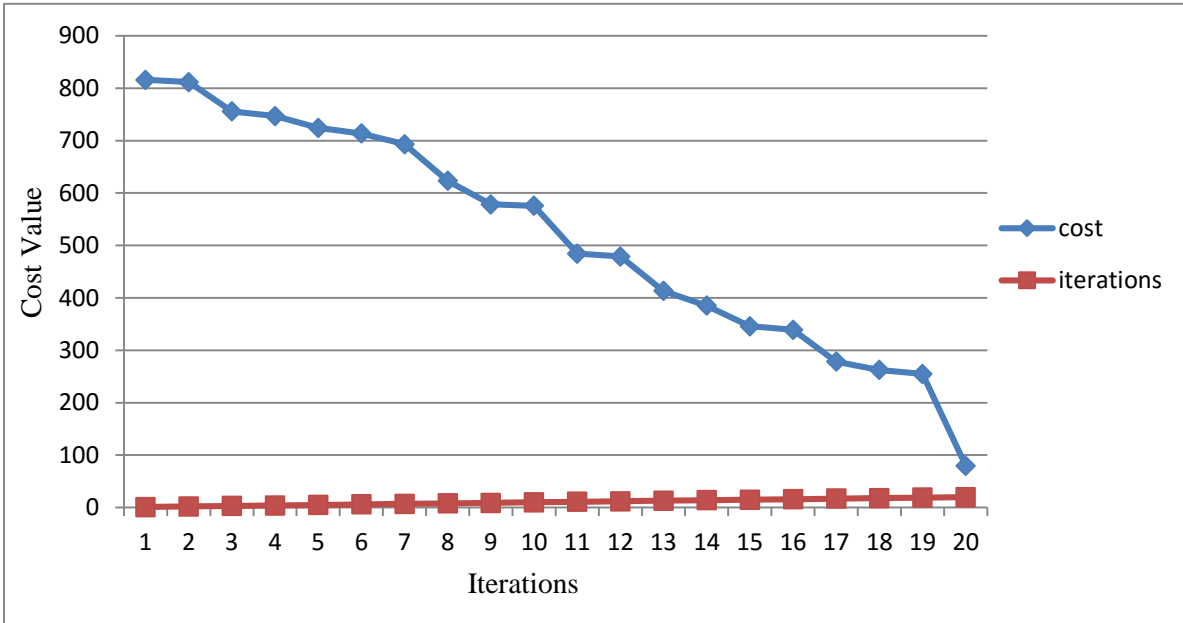


Figure 4.13: Graph Showing the Variation of Function $f = 30.5x(1) + 34x(2) \leq 100$ Corresponding to the Number of Iterations

(v) Function 13: Evaluation and Optimization of function $18x(1) + 19.5x(2) \leq 60$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.13 addresses the evaluation and optimization of the function $f = 18x(1) + 19.5x(2) \leq 60$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 24 which further impacts the optimization of the evaluated cost function value. The variable $x(1)$ =processing time of part 1 on machine 5 and $x(2)$ = processing time of part 2 on machine 5. The constraints in the equation are $x(1) > 0$, $x(2) > 0$ and $f < 100$. The mean of iterations for $x(1)$ comes out to be 13.687 and that of $x(2)$ is 13.475. The best cost function value is obtained is 71.733. Figure 4.14 shows the variation of the values of the function $f = 18x(1) + 19.5x(2) \leq 60$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 805.804 is at first iteration and there after optimized to 71.733 at the last. The minimum value of the function can be seen on the 20th iteration. The dots in the graph

represents, the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

Table 4.13: Evaluation and Optimization of Function $18x(1)+19.5x(2)\leq 60$ Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost Function Value
Iteration 1	21.912	21.096	805.804
Iteration 2	19.631	23.021	802.278
Iteration 3	15.686	23.838	747.205
Iteration 4	21.957	17.507	736.628
Iteration 5	14.819	23.029	715.828
Iteration 6	14.989	22.318	705.026
Iteration 7	23.368	13.420	682.344
Iteration 8	16.005	16.717	614.098
Iteration 9	10.825	19.256	570.364
Iteration 10	17.563	12.803	565.810
Iteration 11	17.960	7.728	474.000
Iteration 12	13.474	11.641	469.539
Iteration 13	2.577	18.472	406.613
Iteration 14	19.501	1.183	374.091
Iteration 15	1.962	15.562	338.786
Iteration 16	1.993	15.192	332.124
Iteration 17	13.639	1.166	268.260
Iteration 18	12.316	1.585	252.609
Iteration 19	12.100	1.693	245.001
Iteration 20	1.474	2.318	71.733
Mean	13.687	13.475	

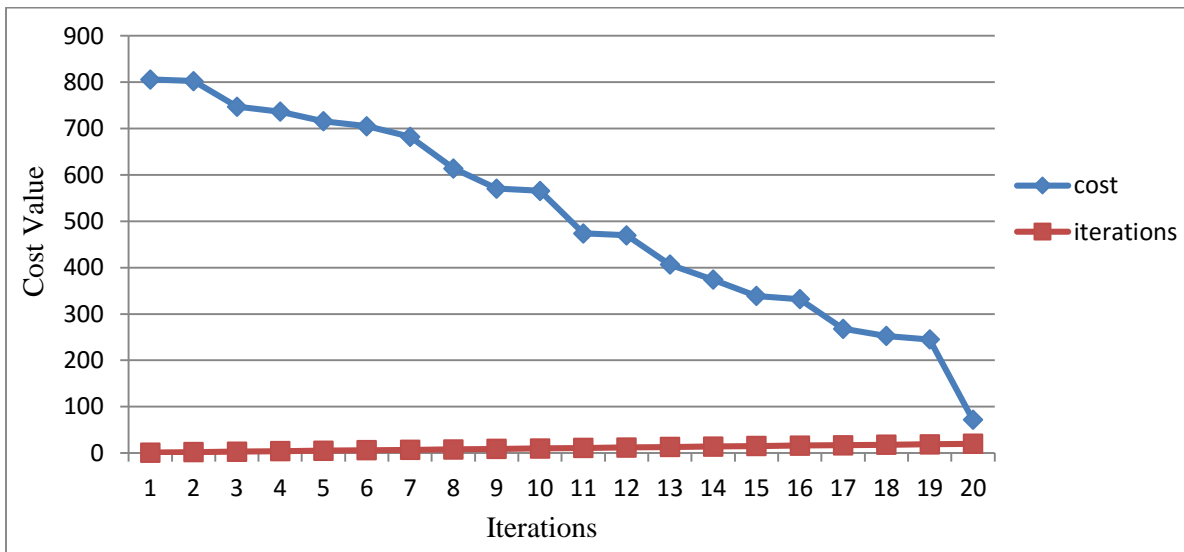


Figure 4.14: Graph Showing the Variation of Function $f=18x(1)+19.5x(2)\leq 60$ Corresponding to the Number of Iterations

(vi) Function 14: Evaluation and Optimization of function $24x(1)+22x(2)\leq 120$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.14 addresses the evaluation and optimization of the function $f=24x(1)+22x(2)\leq 120$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 24 which further impacts the optimization of the evaluated cost function value. The variable $x(1)$ =processing time of part 1 on machine 6 and $x(2)$ = processing time of part 2 on machine 6. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<120$. The mean of iterations for $x(1)$ comes out to be 13.842 and that of $x(2)$ is 11.786.

Table 4.14: Evaluation and Optimization of Function $24x(1)+22x(2)\leq 120$ Using Proposed BBO- TLBO Heuristic

Iterations	$x(1)$	$x(2)$	Cost function value
Iteration 1	22.299	20.375	983.437
Iteration 2	13.991	23.377	850.098
Iteration 3	13.585	22.144	813.223
Iteration 4	22.520	11.614	796.011
Iteration 5	11.272	22.946	775.363
Iteration 6	17.854	12.407	701.468
Iteration 7	9.252	21.442	693.792
Iteration 8	18.968	10.612	688.701
Iteration 9	19.474	7.881	640.798
Iteration 10	22.988	3.151	621.049
Iteration 11	16.797	9.712	616.799
Iteration 12	5.422	21.547	604.330
Iteration 13	16.776	8.556	590.895
Iteration 14	21.832	1.893	565.635
Iteration 15	14.815	6.026	488.146
Iteration 16	6.574	11.509	399.882
Iteration 17	4.979	10.311	346.359
Iteration 18	10.805	1.008	281.527
Iteration 19	4.166	6.885	251.482
Iteration 20	2.461	2.030	103.739
Mean	13.842	11.786	

The best cost function value is obtained is 103.739. Figure 4.15 illustrates the variation of the values of the function $f=24x(1)+22x(2)\leq 120$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 983.437 is at first iteration and there after optimized to 103.739 at the last. The minimum value of the function can be seen on the 20th iteration.

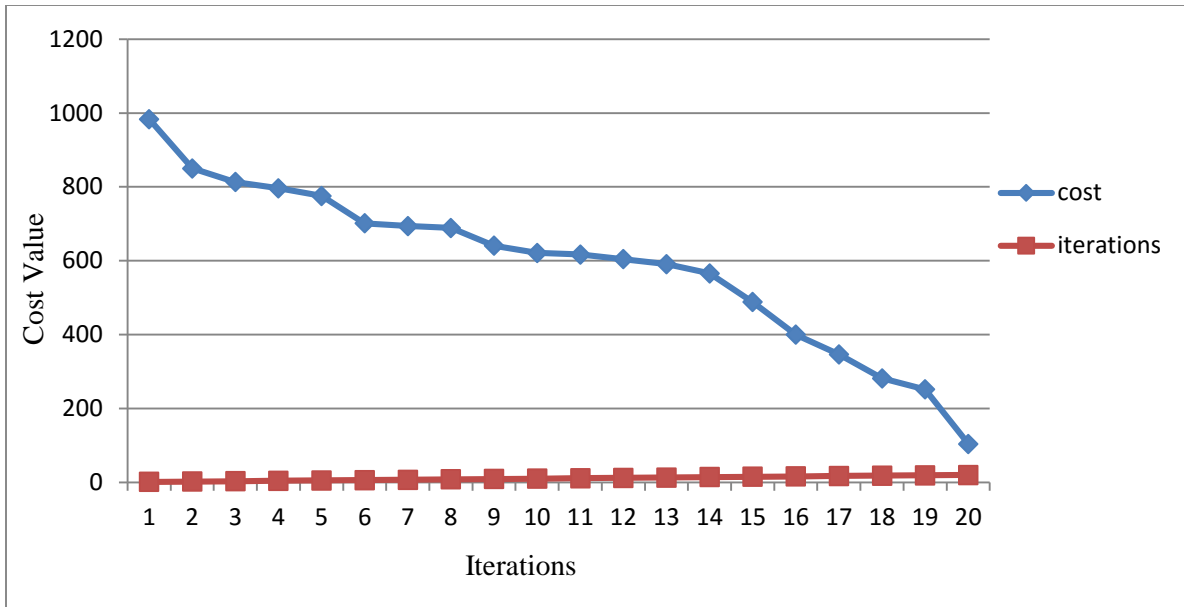


Figure 4.15: Graph Showing the Variation of Function $f=24x(1)+22x(2)\leq 120$ Corresponding to the Number of Iterations

Table 4.15: Evaluation and Optimization of Function $33x(1)+34.5x(2)\leq 100$ Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost function value
Iteration 1	22.782	23.504	1562.736
Iteration 2	18.353	19.186	1267.608
Iteration 3	12.673	22.198	1184.072
Iteration 4	19.154	14.293	1125.242
Iteration 5	21.444	11.000	1087.212
Iteration 6	23.610	8.582	1075.233
Iteration 7	11.388	18.230	1004.778
Iteration 8	12.511	17.035	1000.610
Iteration 9	10.623	18.027	972.533
Iteration 10	6.393	20.176	907.074
Iteration 11	15.490	11.152	895.941
Iteration 12	17.163	9.447	892.332
Iteration 13	2.1791	23.413	879.662
Iteration 14	21.966	1.949	792.152
Iteration 15	20.486	2.629	766.768
Iteration 16	18.584	4.258	760.182
Iteration 17	7.697	13.602	723.302
Iteration 18	11.182	10.086	716.990
Iteration 19	19.004	2.414	710.459
Iteration 20	1.468	6.507	253.799
Mean	14.707	12.840	

(vii) Function 15: Evaluation and Optimization of function $33x(1)+34.5x(2)\leq 100$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.15 addresses the evaluation and optimization of the function $f=33x(1)+34.5x(2)\leq 100$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 24 which further impacts the optimization of the evaluated cost function value. The variable $x(1)$ =processing time of part 1 on machine 7 and $x(2)$ = processing time of part 2 on machine 7. The constraints in the equation are $x(1)>0$, $x(2)> 0$ and $f<100$. The mean of iterations for $x(1)$ comes out to be 14.707 and that of $x(2)$ is 12.840. The best cost function value is obtained is 253.799. Figure 4.16 depicts the variation of the values of the function $f=33x(1)+34.5x(2)\leq 100$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 1562.736 is at first iteration and there after optimized to 253.799 at the last. The minimum value of the function can be seen on the 20th iteration. The dots in the graph represents, the value of function corresponding to the respective iteration. It is to be noted that the rate of optimization is not constant and varies differently between the successive iterations.

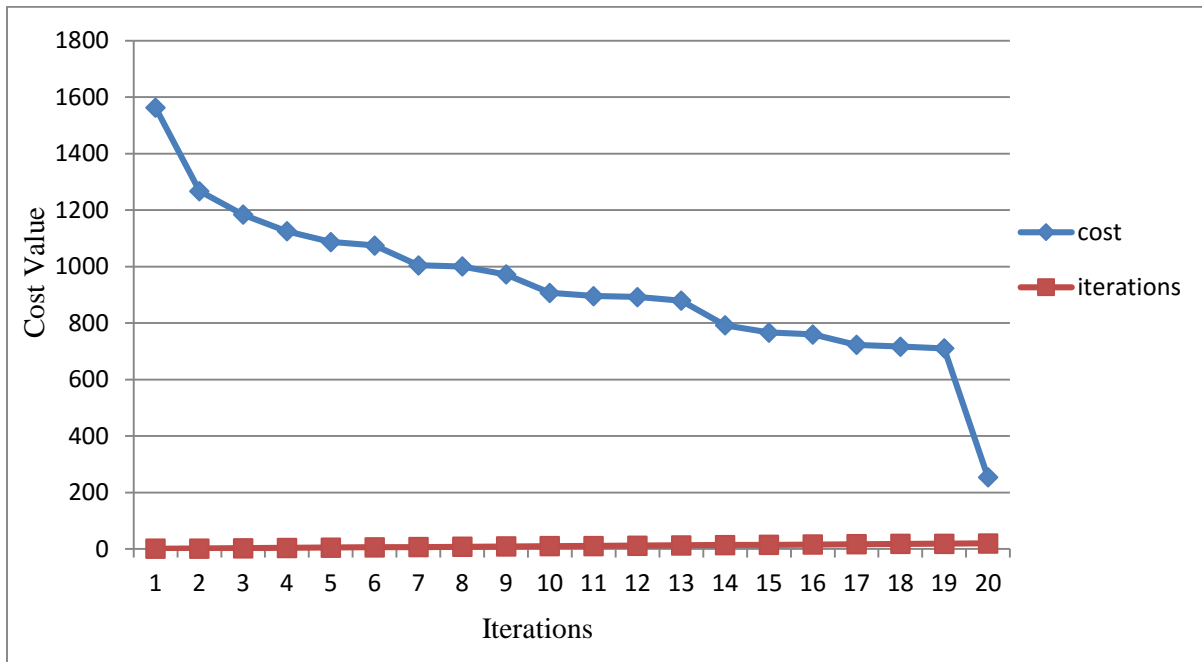


Figure 4.16: Graph Showing the Variation of Function $f=33x(1)+34.5x(2)\leq 100$ Corresponding to the Number of Iterations

(viii) Function 16: Evaluation and Optimization of function $31.5x(1)+30x(2)\leq 120$ for Objective of Minimization of Process Time Using Proposed BBO- TLBO Heuristic

Table 4.16: Evaluation and Optimization of Function $31.5x(1)+30x(2)\leq 120$ Using Proposed BBO- TLBO Heuristic

Iterations	x(1)	x(2)	Cost function value
Iteration 1	23.354	22.278	1404.033
Iteration 2	20.627	20.237	1256.893
Iteration 3	18.586	21.240	1222.705
Iteration 4	13.398	23.213	1118.474
Iteration 5	11.494	23.610	1070.403
Iteration 6	19.990	13.638	1038.686
Iteration 7	9.737	22.452	980.310
Iteration 8	21.256	6.849	875.041
Iteration 9	16.367	11.851	870.986
Iteration 10	11.975	14.755	819.900
Iteration 11	11.150	13.602	759.316
Iteration 12	4.936	19.519	741.072
Iteration 13	16.976	6.696	735.677
Iteration 14	3.900	20.330	732.777
Iteration 15	6.616	15.791	682.181
Iteration 16	13.077	8.298	660.894
Iteration 17	12.799	7.304	622.313
Iteration 18	4.293	15.616	603.746
Iteration 19	10.030	8.868	582.025
Iteration 20	14.453	1.535	501.215
Mean	13.251	14.858	

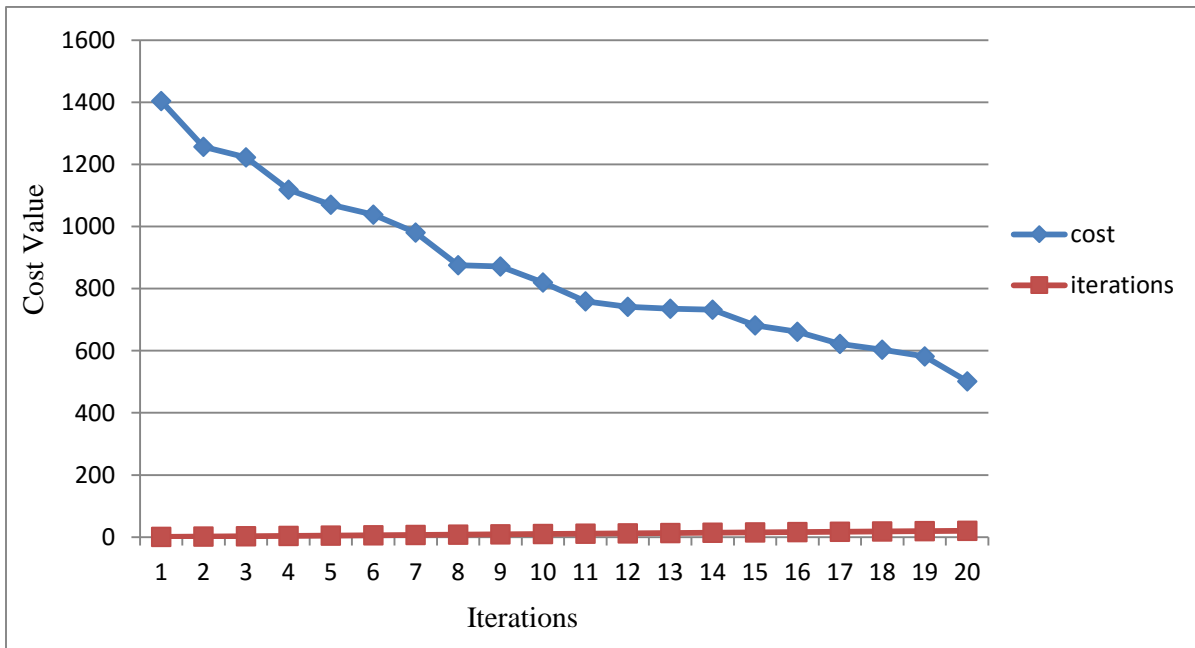


Figure 4.17: Graph Showing the Variation of Function $f=31.5x(1)+30x(2)\leq 120$ Corresponding to the Number of Iterations

Table 4.16 addresses the evaluation and optimization of the function $f=31.5x(1)+30x(2)\leq 120$ by the proposed BBO-TLBO heuristic. In this table, the input variables $x(1)$ and $x(2)$ follows the range between 1 and 24 which further impacts the optimization of the evaluated cost function value. The variable $x(1)$ = processing time of part 1 on machine 8 and $x(2)$ = processing time of part 2 on machine 8. The constraints in the equation are $x(1)>0$, $x(2)>0$ and $f<120$. The mean of iterations for $x(1)$ comes out to be 13.251 and that of $x(2)$ is 14.858. The best cost function value is obtained is 501.215. Figure 4.17 refers the variation of the values of the function $f=31.5x(1)+30x(2)\leq 120$ with the iterations taking place. It is evident from the graph that as the number of iterations are increasing, the value of the function keeps optimizing. The maximum value 1404.033 is at first iteration and there after optimized to 501.215 at the last. The minimum value of the function can be seen on the 20th iteration. The dots in the graph represents, the value of function corresponding to the respective iteration.

4.3 SIMULATION OF SEQUENCE DEPENDENT SETUP TIME ALONG WITH BACLOGGING FOR PART 1 AND PART 2

In scheduling, set-up time makes problem more complex and comes to play when production changeover is required between the different jobs, taking different amount of time to set-up on the machine before starting the operation. There are two type of structures; simple, in which set-up is independent of sequences and decisions for previous times, and complex, in which set-up time is dependent on both the factors. Backlog is the uncompleted, unprocessed work for a specified time or jobs in the process of completion. It implies to the workload, which is beyond the capacity of the production system. The factor on which it depends is waiting time more the waiting time lower is the backlogging rate. Partial backlogging is a situation where the demand of a product met from other sources where as in full backlogging, demand remains unfulfilled until the next order.

4.3.1 Simulation of Set up and Backlogging for Part 1

The manufacturing of part 1 that is ring carrier piston is done on the eight different machines encountering different times on different machines. Table 4.17 shows the set up time, process time, backlog time, cycle tie for the part 1 on the various machines and the reduced cycle time, obtained by the implementation of BBO-TLBO heuristic. On machine M1 the set up time is 2 min, backlog time is 4 min, cycle time is 17 min and reduced to 13.189 min by using the

developed heuristic. On machine M2 the set up time is 1.5 min, backlog time is 5 min, cycle time is 18 min and reduced to 11.442 min by using the developed heuristic. On machine M3 the set up time is 2 min, backlog time is 5 min, cycle time is 19.5 min and reduced to 13.013 min by using the developed heuristic. On machine M4 the set up time is 3 min, backlog time is 8 min, cycle time is 30.5 min and reduced to 15.083 min by using the developed heuristic. On machine M5 the set up time is 2 min, backlog time is 4.5 min, cycle time is 18 min and reduced to 13.687 min by using the developed heuristic. On machine M6 the set up time is 2 min, backlog time is 6.5 min, cycle time is 24 min and reduced to 13.842 min by using the developed heuristic. On machine M7 the set up time is 2.5 min, backlog time is 11 min, cycle time is 33 min and reduced to 14.707 min by using the developed heuristic. On machine M8 the set up time is 3.5 min, backlog time is 10 min, cycle time is 31.5 min and reduced to 13.25 min by using the developed heuristic.

Table 4.17: Depiction of the Various Times (in min) of Part 1 on Various Machines

Part 1	M1	M2	M3	M4	M5	M6	M7	M8
Set up Time	2	1.5	2	3	2	2	2.5	3.5
Process Time	11	11	12	19	11	15	19	15
Backlog Time	4	5	5	8	4.5	6.5	11	10
Cycle Time	17	18	19.5	30.5	18	24	33	31.5
Reduced Cycle Time	13.189	11.442	13.013	15.083	13.687	13.842	14.707	13.251

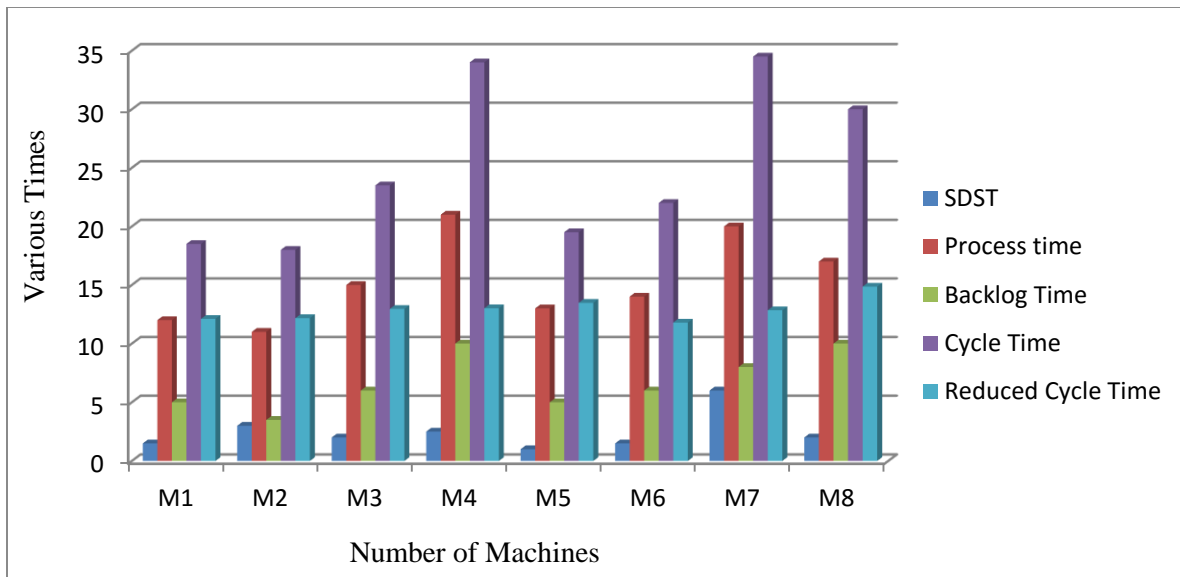


Figure 4.18: Depiction of Set up Time, Process Time, Backlog Time, Cycle Time and Reduced Cycle Time for Part 1

Figure 4.18 depicts various times taken by part 1 on the eight machines in the form of colored bars. It is evident from the graph that most time reduced is on the machine M7 which reduces the cycle time from 33 min to 14.707 min and the lowest reduction in the cycle time is on the machine M1 from 17 min to 13.189 min. The reduction in the cycle time for other machines lies between 3.811 min to 18.293 min. The clear comparative visualization of different times on the respective machines can be easily seen with the different colored bars.

4.3.2 Simulation of Set up time and Backlogging for Part 2

The manufacturing of part 1 that is four stroke bi wheeler piston rings is done on the eight different machines which takes different times for the respective operations to be executed.

Table 4.18: Depiction of the Various Times of Part 2 on Various Machines

Part 2	M1	M2	M3	M4	M5	M6	M7	M8
Set up time	1.5	3	2	2.5	1	1.5	6	2
Process time	12	11	15	21	13	14	20	17
Backlog Time	5	3.5	6	10	5	6	8	10
Cycle Time	18.5	18	23.5	34	19.5	22	34.5	30
Reduced Cycle Time	12.106	12.169	12.955	13.012	13.475	11.786	12.840	14.858

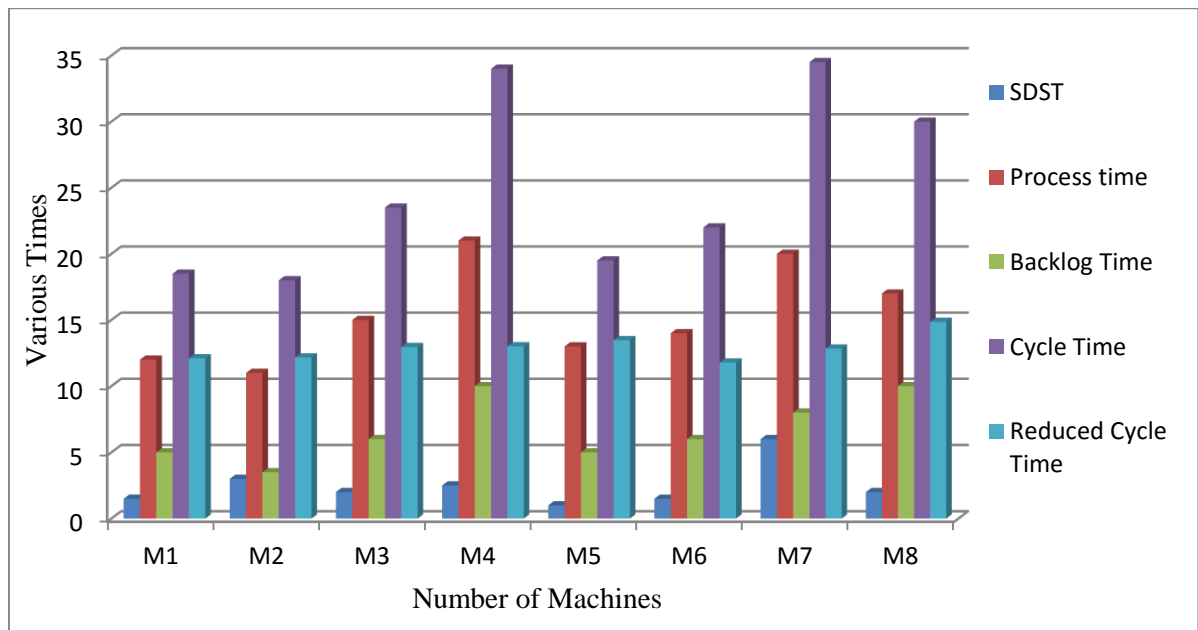


Figure 4.19: Depiction of Set up Time, Process Time, Backlog Time, Cycle Time and Reduced Cycle Time for Part 2

Table 4.18 shows the set up time, process time, backlog time, cycle time for part 1 on the various machines and the reduced cycle time, obtained by the implementation of BBO-TLBO

heuristic. On machine M1 the set up time is 1.5 min, backlog time is 5 min, cycle time is 18.5 min and reduced to 12.106 min by using the developed heuristic. On machine M2 the set up time is 3 min, backlog time is 3.5 min, cycle time is 18 min and reduced to 12.169 min by using the developed heuristic. On machine M3 the set up time is 2 min, backlog time is 6 min, cycle time is 23.5 min and reduced to 12.955 min by using the developed heuristic. On machine M4 the set up time is 2.5 min, backlog time is 10 min, cycle time is 34 min and reduced to 13.012 min by using the developed heuristic. On machine M5 the set up time is 1 min, backlog time is 5 min, cycle time is 19.5 min and reduced to 13.475 min by using the developed heuristic. On machine M6 the set up time is 1.5 min, backlog time is 6 min, cycle time is 22 min and reduced to 11.786 min by using the developed heuristic. On machine M7, set up time is 6 min, backlog time is 8 min, cycle time is 34.5 min and reduced to 12.240 min by using the developed heuristic. On machine M8 the set up time is 2 min, backlog time is 10 min, cycle time is 30 min and reduced to 14.858 min by using the developed heuristic. Figure 4.19 shows the various times taken by part 1 on the eight machines in the form of colored bars. It is evident from the graph that most time reduced is on machine M7 which reduces the cycle time from 34.5 min to 12.840 min and the lowest reduction in the cycle time is on the machine M2 from 18 min to 12.169 min. The reduction in the cycle time for other machines lies between 5.831 min to 21.66 min. The clear comparative visualization of different times on the respective machines can be easily seen with the different colored bars.

4.4 VALIDATION OF RESULTS GENERATED BY PROPOSED BBO-TLBO HEURISTIC FOR MINIMIZATION OF PROCESS TIME AND CYCLE TIME

Validation is the process of certifying the produced results that if they are better than the actual data used or not. This process has an advantage to ascertaining the quality of the results produced. The validation of the results produced by the proposed BBO-TLBO heuristic is compared with the actual data used to form the various functions. These functions were supposed to be minimized by the heuristic developed. The objectives namely process time and cycle time undertaken for the both parts and all the machines from 1 to 8 will be validated separately to ensure the quality of results. The below subsections shows the difference between the actual and the generated results and also depicts the quality of the results produced.

4.4.1 Validation of Results for the Objective of Minimization of Process Time

Table 4.19 and Table 4.20 show the actual data used to form the various objective functions and the results generated from the proposed heuristic, respectively. Now, as there

Table 4.19: Depiction of the Actual Data of Process Time (in minutes) from Industry

	M1	M2	M3	M4	M5	M6	M7	M8
Part 1	11	11	12	19	11	15	19	15
Part 2	12	11	15	21	13	14	24	17

Table 4.20: Depiction of Results Generated (in minutes) from Proposed BBO-TLBO Heuristic

	M1	M2	M3	M4	M5	M6	M7	M8
Part 1	6.365	8.252	8.816	8.460	9.572	8.723	7.613	9.343
Part 2	7.402	7.266	7.565	7.564	8.311	8.592	8.343	8.206

are two parts namely Part 1 and Part 2, which were worked upon eight different machines namely M1 to M8, so process time will be validated for both the parts separately.

4.4.1.1 Validation of generated results for Process time (in minutes) for Part 1 on all the machines

By comparing the actual data of part 1 and the results obtained from the heuristic BBO-TLBO, it is evident from the table 4.21, that the processing time of the part 1 on all the machines has been reduced. Table 4.21 displays the reduced values of actual process time of every machine from M1 to M8 for Part 1. Figure 4.20 displays the data of Table 4.21 in the form of bars. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The red bars displaying the results obtained from BBO-TLBO heuristic are considerably less than the actual value of the processing time. Table 4.22 contains the results obtained from the proposed heuristic and the optimized time. The optimized time is the difference between the actual data and the results obtained. The optimized time shows the quality of the results and the effectiveness of the heuristic in minimizing the values processing time.

Table 4.21: Depiction of Actual Process Time and Results by BBO-TLBO Heuristic (in min)

Machines	M1	M2	M3	M4	M5	M6	M7	M8
Actual time of Part 1	11	11	12	19	11	15	19	15
Results obtained from BBO-TLBO Heuristic for Part 1	6.365	8.252	8.816	8.460	9.572	8.723	7.613	9.343

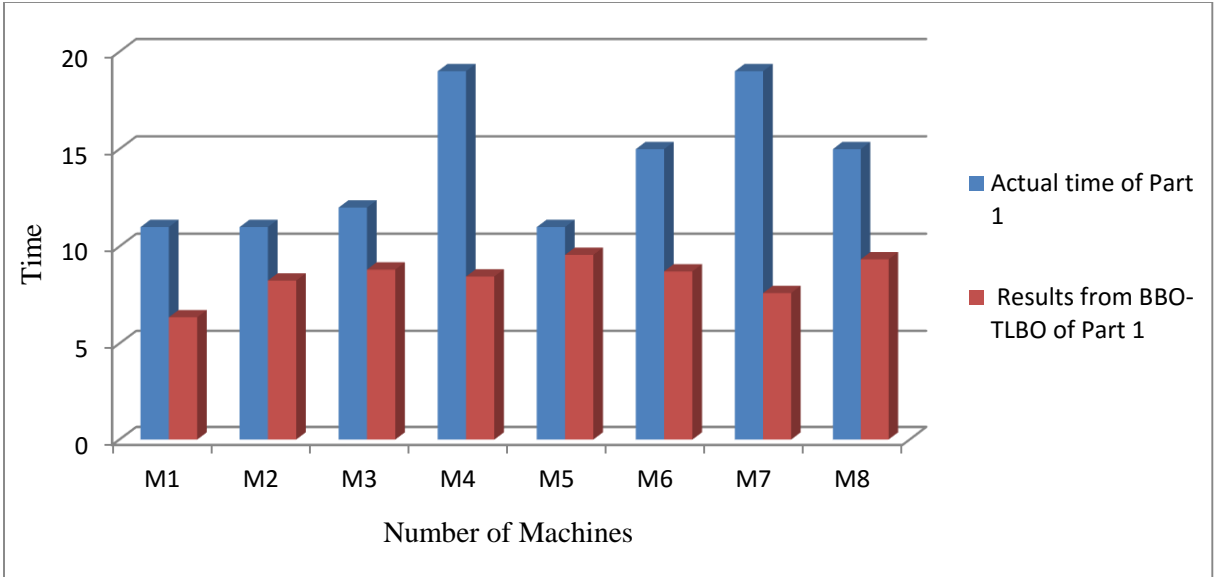


Figure 4.20: Graph Displaying the Actual Process time and Results by BBO-TLBO Heuristic

Table 4.22: Depiction of Results by BBO-TLBO Heuristic and Optimized Time (in min)

Machines	M1	M2	M3	M4	M5	M6	M7	M8
Results from BBO-TLBO of Part 1	6.365	8.252	8.816	8.460	9.572	8.723	7.613	9.343
Optimized Process Time data for Part 1	4.635	2.475	3.184	10.540	1.428	6.277	11.387	5.657

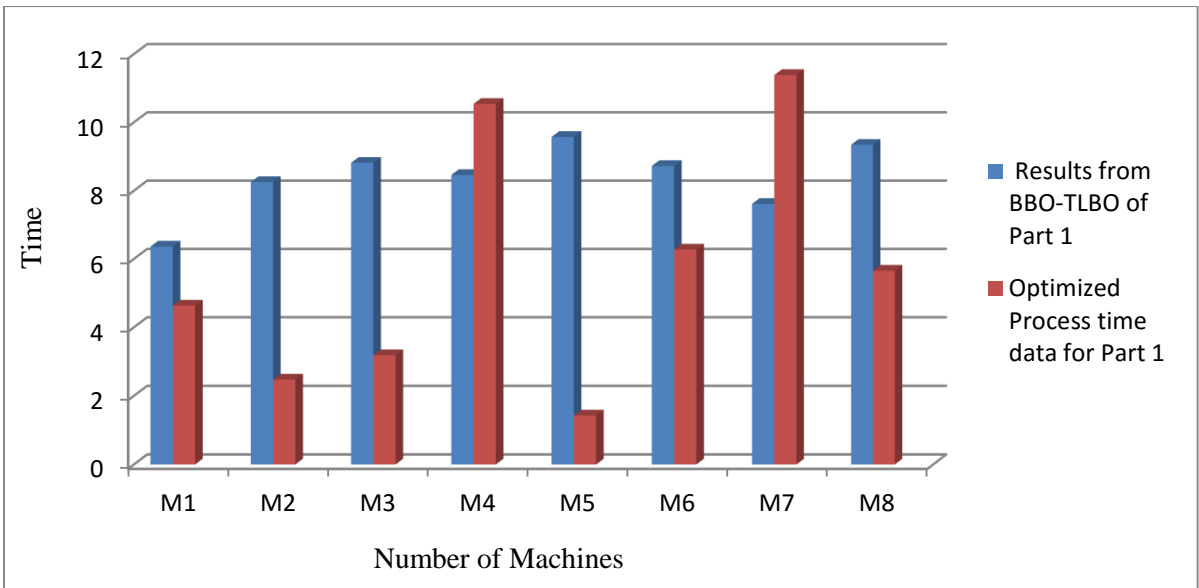


Figure 4.21: Graph Showing Results of BBO-TLBO Heuristic and Optimized Time (in min)

Figure 4.21 displays the results obtained from heuristic and the optimized value of the process time for Part 1 on all the machines. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The minimum optimized time is 1.428 min on machine 5 and the maximum is 11.387 min on the machine 7. This is evident from the red bars of the graph that the heuristic is able to optimize a significantly on all the machines for part 1.

Table 4.23: Depiction of Actual Process time and Optimized time (in min)

Machines	M1	M2	M3	M4	M5	M6	M7	M8
Actual time of Part 1	11	11	12	19	11	15	19	15
Optimized Process time data for Part 1	4.635	2.475	3.184	10.54	1.428	6.277	11.387	5.657

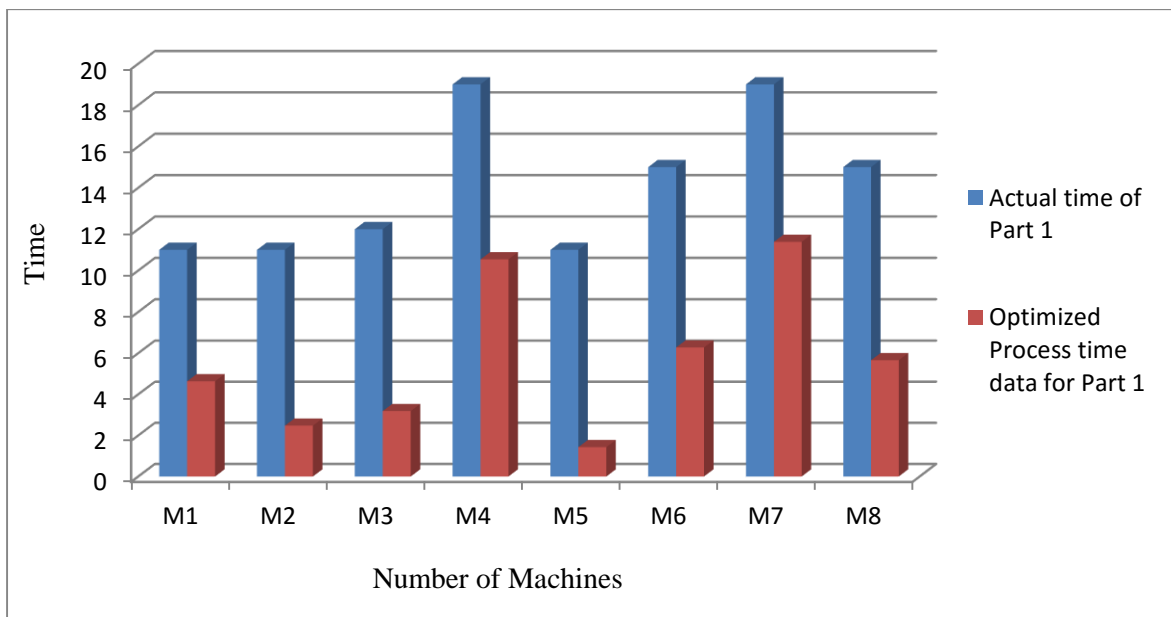


Figure 4.22: Graph Displaying the Actual Process Time and Optimized Time (in min)

Table 4.23 shows the actual time spent by the Part 1 in the industry and the optimized time obtained from the proposed heuristic. It is clearly evident from the table that the proposed heuristic has been able to optimize the process time for Part 1 on all the machines. Figure 4.22 displays the actual processing time and the optimized time for Part 1 on all the machines. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The blue bars shown as the actual time are much bigger than the red bars showing the optimized time, but are able to significantly cut the slack of the processing time. The most processing time is employed to machine M4 and M7, and the most optimized time 11.387 min is on

machine M7. Table 4.22 displays the combined set of actual data, results by heuristic BBO-TLBO and the optimized data for Part 1 corresponding to each machine. The minimum optimized time is on the machine M5 with 1.428 min and the maximum is on the machine M7 with 11.387 minutes.

Table 4.24: Depiction of Cumulative Validation of Actual Data, Results by BBO-TLBO and Optimized Data for Part 1

Machines	M1	M2	M3	M4	M5	M6	M7	M8
Actual time of Part 1	11	11	12	19	11	15	19	15
Results from BBO-TLBO of Part 1	6.365	8.252	8.816	8.46	9.572	8.723	7.63	9.343
Optimized Process time data for Part 1	4.635	2.475	3.184	10.54	1.428	6.277	11.387	5.657

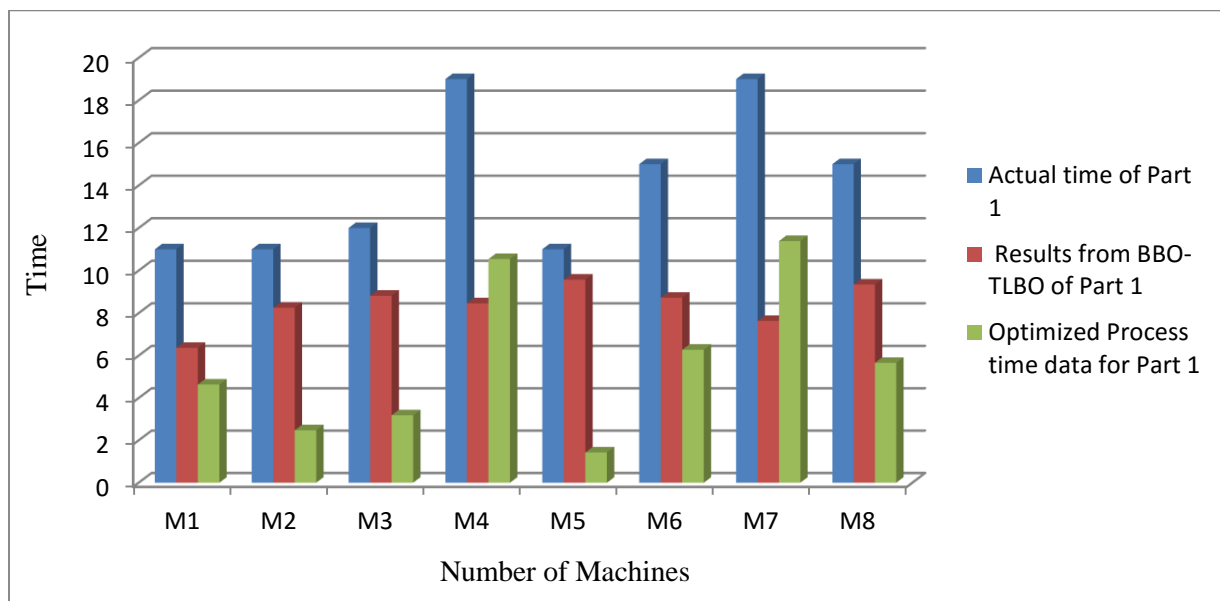


Figure 4.23: Graph Showing the Cumulative Validation of Actual data, results by BBO-TLBO and Optimized Data for Part 1

The values of the optimized times for all the other machines varies between these two. Figure 4.23 cumulatively displays the actual process time, results from heuristic and the optimized time for part 1. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The blue bars denote the actual process time on the respective machines holds the highest peaks, but the red bars showing the results from heuristic are significant and optimizes the process time on every machine.

4.4.1.2 Validation of generated results for Process time (in minutes) for Part 2 on all the machines.

By comparing the actual data of part 1 and the results obtained from the heuristic BBO-TLBO, it is evident from the Table 4.25, that the processing time of the Part 1 on all the machines has been reduced. Table 4.25 displays the reduced values of actual process time of every machine from M1 to M8 for part 2.

Table 4.25: Depiction of Actual Process Time and Results by BBO-TLBO Heuristic (in min)

Machines	M1	M2	M3	M4	M5	M6	M7	M8
Actual Process time of Part 2	12	11	15	21	13	14	24	17
Results from BBO-TLBO of Part 2	7.402	7.266	7.565	7.564	8.311	8.592	8.343	8.206

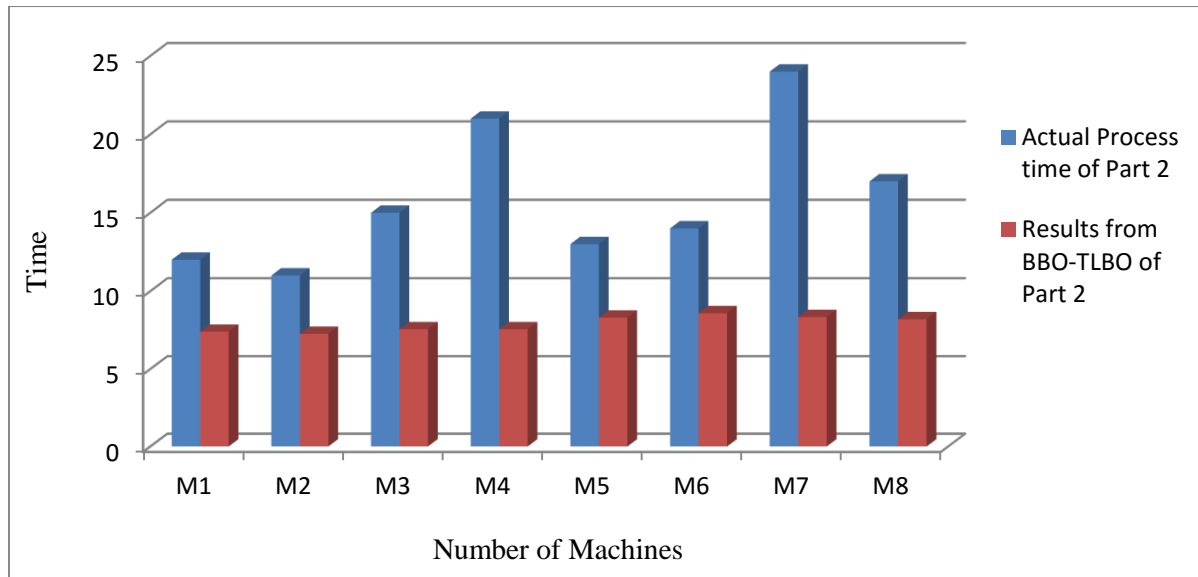


Figure 4.24: Graph Showing the Actual Process Time and Results by BBO-TLBO Heuristic (in min)

Table 4.26: Depiction of Results by BBO-TLBO Heuristic and Optimized Time (in min)

	M1	M2	M3	M4	M5	M6	M7	M8
Results from BBO-TLBO of Part 1	7.402	7.266	7.565	7.564	8.311	8.592	8.343	8.206
Optimized Time for Part 2	4.598	3.734	7.435	13.436	4.689	5.408	15.657	8.794

Figure 4.24 displays the data of Table 4.21 in the form of bars. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The red bars displaying the results obtained from BBO-TLBO heuristic are considerably less than the actual value of the

processing time. Table 4.26 contains the results obtained from the proposed heuristic and the optimized time. The optimized time is the difference between the actual data and the results obtained. The optimized time shows the quality of the results and the effectiveness of the heuristic in minimizing the values processing time. Figure 4.25 depicts the results obtained from heuristic and the optimized value of the process time for part 1 on all the machines. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The minimum optimized time is 3.734 min on machine M2 and the maximum is 15.657 min on the machine M7. This is evident from the red bars of the graph that the heuristic is able to optimize a significantly on all the machines for part 1.

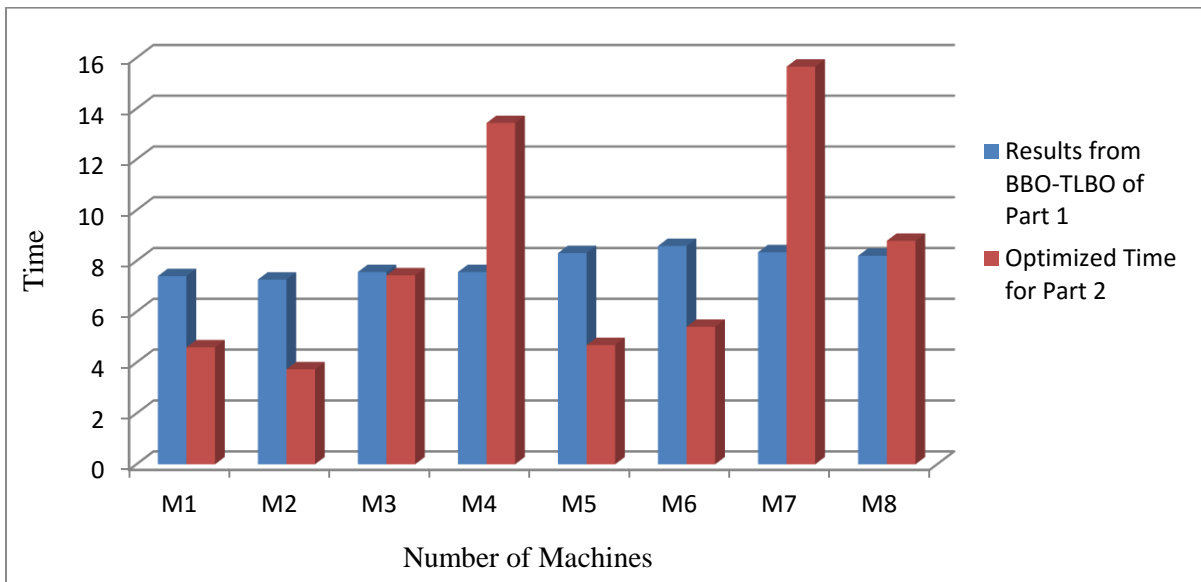


Figure 4.25: Graph Showing Results by BBO-TLBO Heuristic and Optimized Process Time (in min)

Table 4.27 shows the actual time spent by the part 2 in the industry and the optimized time obtained from the proposed heuristic. It is clearly evident from the table that the proposed heuristic has been able to optimize the process time for part 2 on all the machines.

Table 4.27: Depiction of Actual Process Time and Optimized time (in min)

Machines	M1	M2	M3	M4	M5	M6	M7	M8
Actual Process time of Part 2	12	11	15	21	13	14	24	17
Optimized Process Time for Part 2	4.598	3.734	7.435	13.436	4.689	5.408	15.657	8.794

Figure 4.26 displays the actual processing time and the optimized time for part 1 on all the machines. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The blue bars shown as the actual time are much bigger than the red bars

showing the optimized time, but are able to significantly cut the slack of the processing time. The most processing time 21 min is employed to machine M4, and the most optimized time 15.657 min is on machine M7.

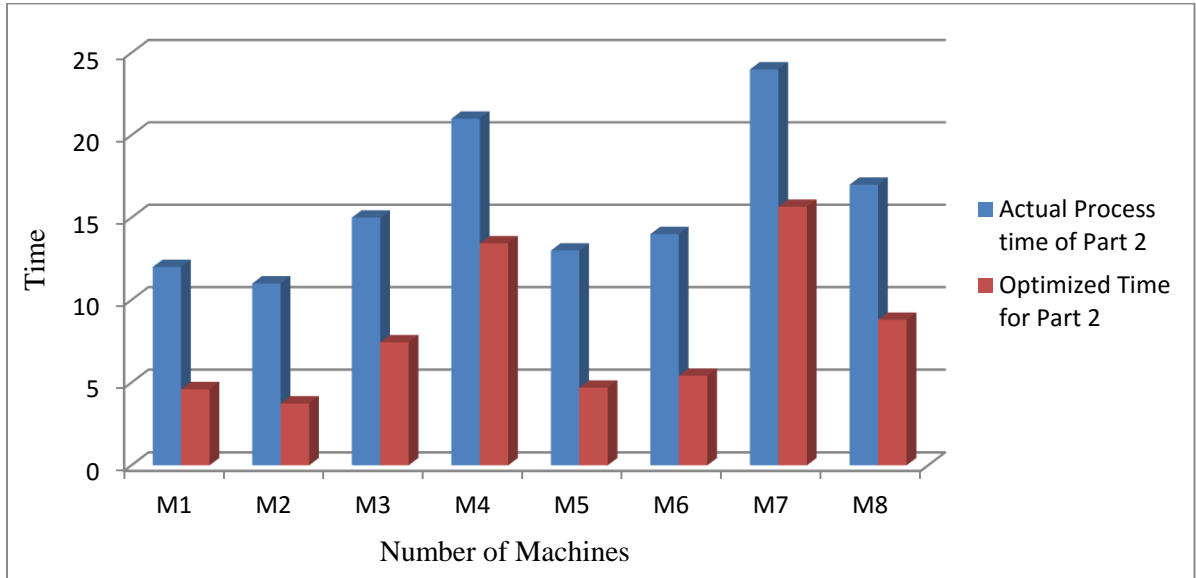


Figure 4.26: Graph Showing Actual Process Time and Optimized Process Time (in min)

Table 4.28 displays the combined set of actual data, results by heuristic BBO-TLBO and the optimized data for part 2 corresponding to each machine. The minimum optimized cycle time is on the machine M2 with 3.734 min and the maximum is on the machine M7 with 15.657 min. The values of the optimized times for all the other machines varies between these two. Figure 4.27 cumulatively depicts the actual process time, results from heuristic and the optimized time for part 2. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The blue bars denote the actual process time on the respective machines holds the highest peaks, but the red bars showing the results from heuristic are significant and optimizes the process time on every machine.

Table 4.28: Depiction of Cumulative Validation of Actual data, results by BBO-TLBO and Optimized Data for Part 2

Machines	M1	M2	M3	M4	M5	M6	M7	M8
Actual Process time of Part 2	12	11	15	21	13	14	24	17
Results from BBO-TLBO of Part 2	7.402	7.266	7.565	7.564	8.311	8.592	8.343	8.206
Optimized Time for Part 2	4.598	3.734	7.435	13.436	4.689	5.408	15.657	8.794

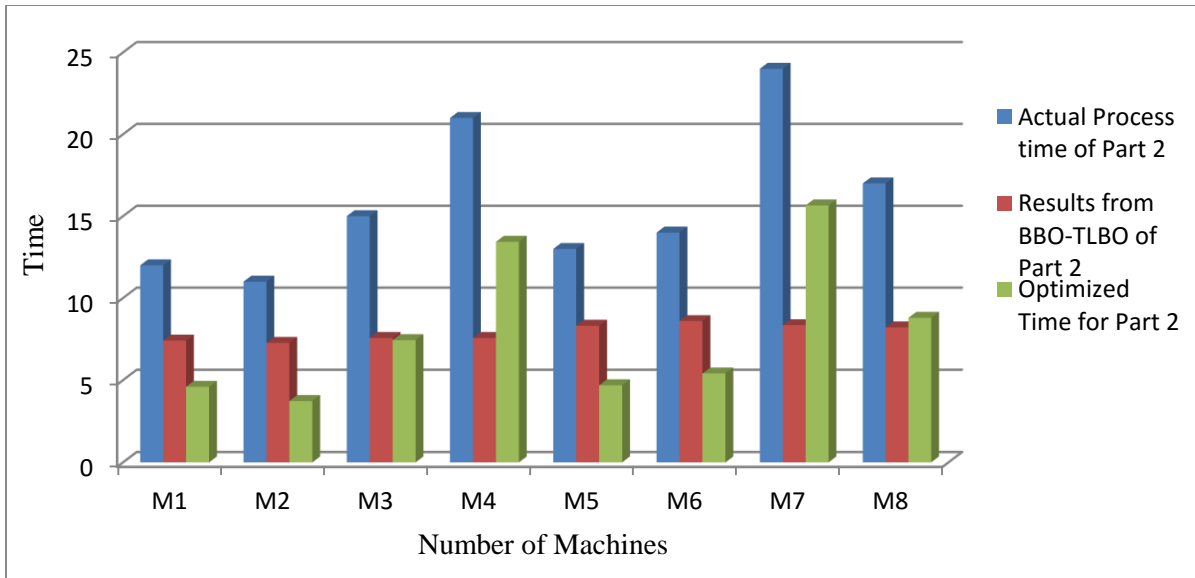


Figure 4.27: Graph Showing the Cumulative Validation of Actual Data, Results by BBO-TLBO and Optimized Data for Part 2

4.4.2 Validation of Results for the Objective of Minimization of Cycle Time

Table 4.29 and Table 4.30 show the actual data used to form the various objective functions and the results generated from the proposed heuristic, respectively. Now, as there are two parts namely part 1 and part 2, which were worked upon eight different machines namely M1 to M8, so cycle time will be validated for both the parts separately.

Table 4.29: Depiction of the Actual Data of Cycle Time (in minutes) from Industry

	M1	M2	M3	M4	M5	M6	M7	M8
Part 1	17	18	19.5	30.5	18	24	33	31.5
Part 2	18.5	18	23.5	34	19.5	22	34.5	30

Table 4.30: Depiction of the Results Generated (in minutes) from Proposed BBO-TLBO Heuristic

	M1	M2	M3	M4	M5	M6	M7	M8
Part 1	13.139	11.426	13.013	15.083	13.687	13.842	14.707	13.251
Part 2	12.106	12.169	12.955	13.012	13.475	11.786	12.840	14.858

4.4.2.1 Validation of generated results for cycle time (in minutes) for part 1 on all the machines. By comparing the actual data of part 1 and the results obtained from the heuristic BBO-TLBO, it is evident from the Table 4.31, that the cycle time of the part 1 on all the machines has been reduced. Table 4.31 displays the reduced values of actual process time of every machine from M1 to M8 for part 1.

Table 4.31: Depiction of Actual Data of cycle Time and Results Generated (in minutes) from Proposed BBO-TLBO Heuristic for Part 1

	M1	M2	M3	M4	M5	M6	M7	M8
Actual data of cycle time Part 1	17	18	19.5	30.5	18	24	33	31.5
Results produced by BBO-TLBO for Part 1	13.139	11.426	13.013	15.083	13.687	13.842	14.707	13.251

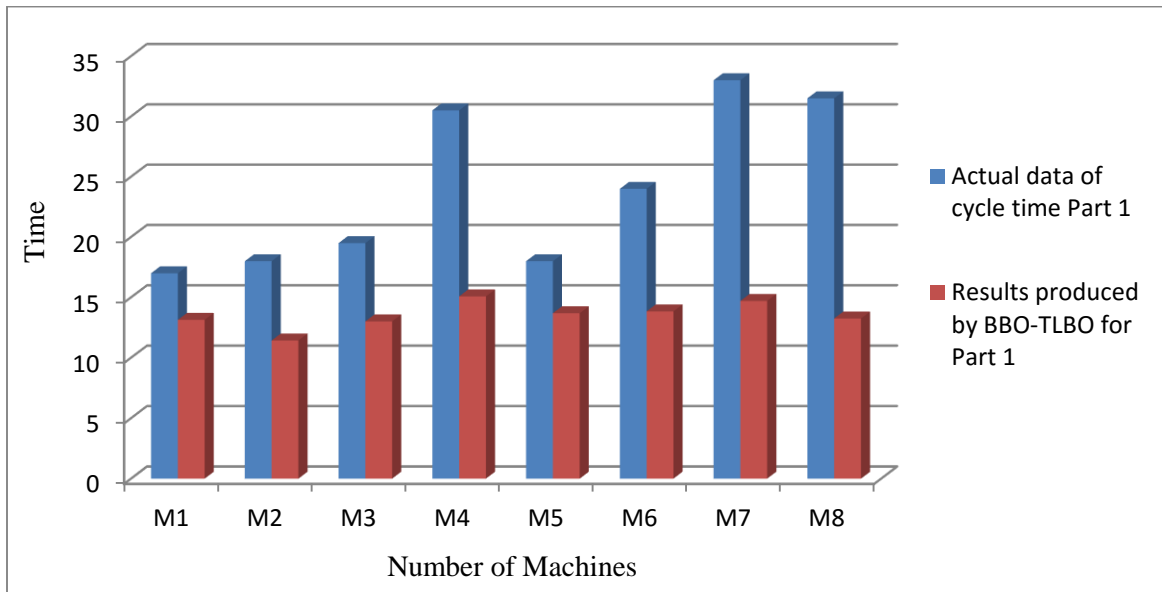


Figure 4.28: Graph Displaying the Actual Data of Cycle Time and Results Generated (in minutes) from Proposed BBO-TLBO Heuristic for Part 1

Figure 4.28 displays the above data of Table 4.29 in the form of bars. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The red bars displaying the results obtained from BBO-TLBO heuristic are considerably less than the actual value of the processing time.

Table 4.32: Depiction of Results Generated (in minutes) from Proposed BBO-TLBO Heuristic and the Optimized data for Part 1

	M1	M2	M3	M4	M5	M6	M7	M8
Results produced by BBO-TLBO for Part 1	13.139	11.426	13.013	15.083	13.687	13.842	14.707	13.251
Optimized Data for part 1	3.861	6.574	6.487	15.417	4.313	10.158	18.293	18.249

Table 4.32 contains the results obtained from the proposed heuristic and the optimized time. The optimized time is the difference between the actual data and the results obtained. The optimized time shows the quality of the results and the effectiveness of the heuristic in minimizing the values of cycle time. Figure 4.29 display the results obtained from heuristic and the optimized value of the cycle time for part 1 on all the machines. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The minimum optimized time is 3.861 min on machine M1 and the maximum is 18.293 min on the machine M7. This is evident from the red bars of the graph that the heuristic is able to optimize a significantly on all the machines for part 1.

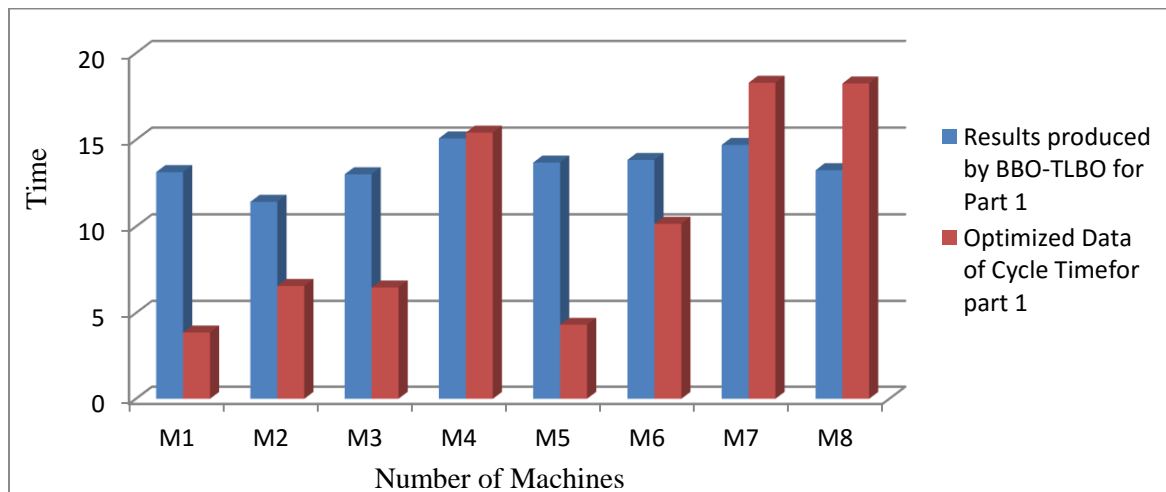


Figure 4.29: Graph Showing the Results Generated (in minutes) from Proposed BBO-TLBO Heuristic and the Optimized Data for Part 1

Table 4.33: Depiction of Actual Data and Optimized Data for Part 1

	M1	M2	M3	M4	M5	M6	M7	M8
Actual data of cycle time Part 1	17	18	19.5	30.5	18	24	33	31.5
Optimized Data for Part 1	3.861	6.574	6.487	15.417	4.313	10.158	18.293	18.249

Table 4.33 depicts the actual time spent by the part 1 in the industry and the optimized time obtained from the proposed heuristic. It is clearly evident from the table that the proposed heuristic has been able to optimize the cycle time for part 1 on all the machines. Figure 4.30 displays the actual cycle time and the optimized time for part 1 on all the machines. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The

blue bars shown as the actual time are much bigger than the red bars showing the optimized time, but are able to significantly cut the slack of the processing time.

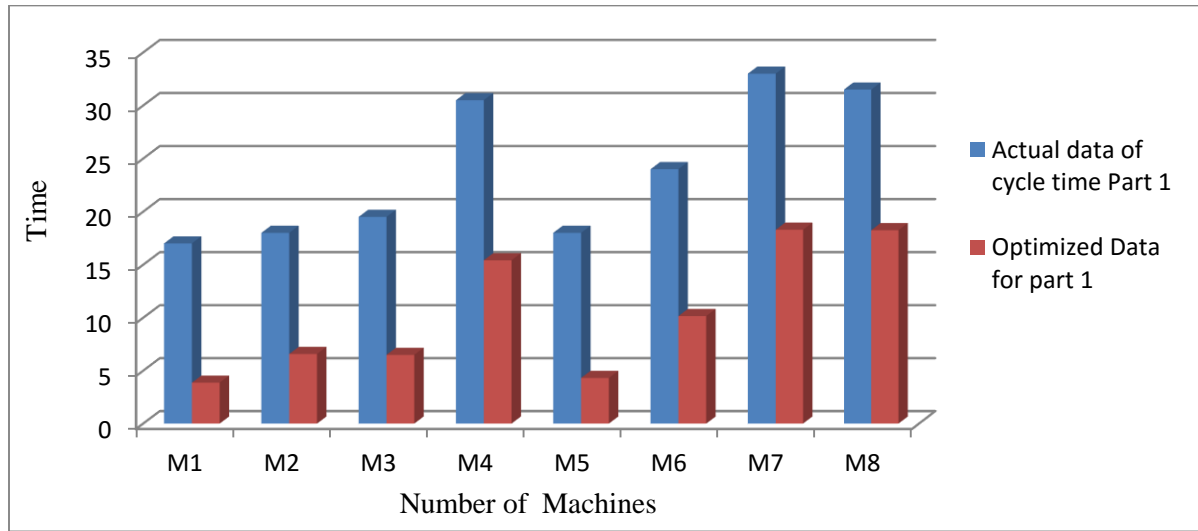


Figure 4.30: Graph Showing Actual Data and Optimized Data for Part 1

The most cycle time is employed to machine M8, and the most optimized time 18.293 min is on machine M7. Table 4.34 depicts the combined set of actual data, results by heuristic BBO-TLBO and the optimized data for Part 1 corresponding to each machine. The minimum optimized time is on the machine M1 with 3.861 min and the maximum is on the machine M7 with 18.293 min. The values of the optimized times for all the other machines varies between these two. Figure 4.31 cumulatively displays actual cycle time, results from BBO-TLBO heuristic and the optimized time for part 1. The x-axis of the graph displays the number of machines and y-axis displays the cycle time.

Table 4.34: Depiction of Actual Data, Results Generated from Proposed BBO-TLBO Heuristic and Optimized Data (in minutes) for Part 1

	M1	M2	M3	M4	M5	M6	M7	M8
Actual data of cycle time Part 1	17	18	19.5	30.5	18	24	33	31.5
Results produced by BBO-TLBO for Part 1	13.139	11.426	13.013	15.083	13.687	13.842	14.707	13.251
Optimized Data for Part 1	3.861	6.574	6.487	15.417	4.313	10.158	18.293	18.249

The blue bars denote the actual process time on the various machines, holds the highest peaks, but the red bars showing the results from heuristic are significant and optimizes the cycle time on every machine.

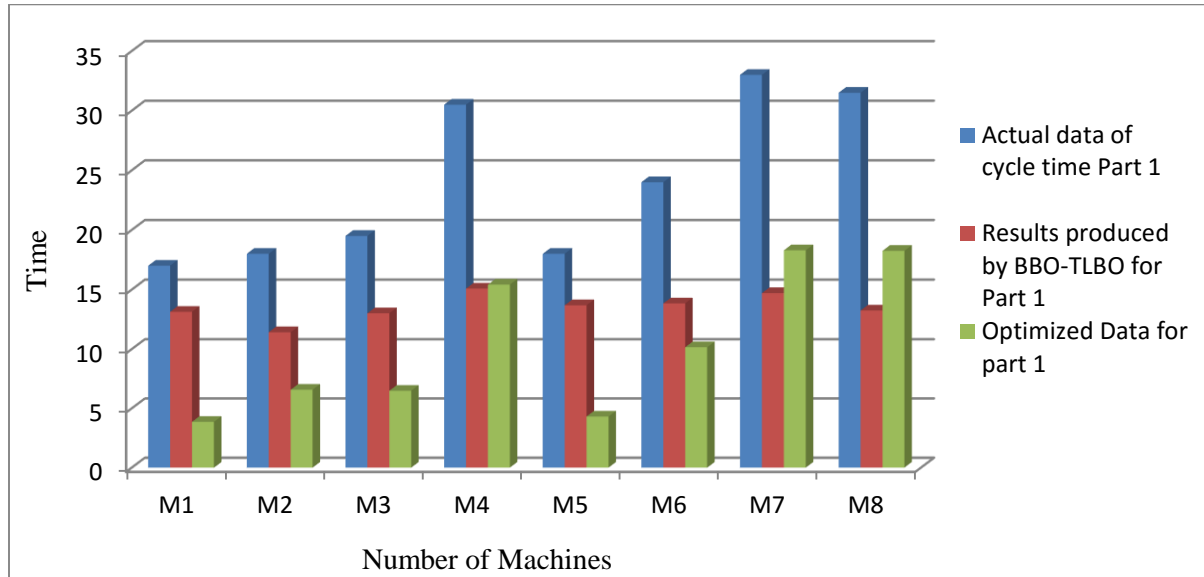


Figure 4.31: Graph Showing the Cumulative Display of Actual Data, Results Generated from Proposed BBO-TLBO Heuristic and Optimized Data (in minutes) for Part 1

4.4.2.2 Validation of generated results for cycle time (in minutes) for part 2 on all the machines By comparing actual data of part 2 and the results obtained from the heuristic BBO – TLBO, it is evident from table 4.35, that cycle time of the part 2 on all the machines has been reduced. Table 4.35 displays the reduced values of actual cycle time of every machine from M1 to M8 for part 1.

Table 4.35: Depiction of Actual Data of cycle Time and Results Generated (in minutes) from Proposed BBO-TLBO Heuristic for Part 2

	M1	M2	M3	M4	M5	M6	M7	M8
Actual Cycle Time data Part 2	17	18	19.5	30.5	18	24	33	31.5
Results from BBO-TLBO for part 2	12.106	12.169	12.955	13.012	13.475	11.786	12.840	14.858

Figure 4.32 displays the above data of table 4.33 in the form of bars. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The red bars

displaying the results obtained from BBO-TLBO heuristic are considerably less than the actual value of the cycle time.

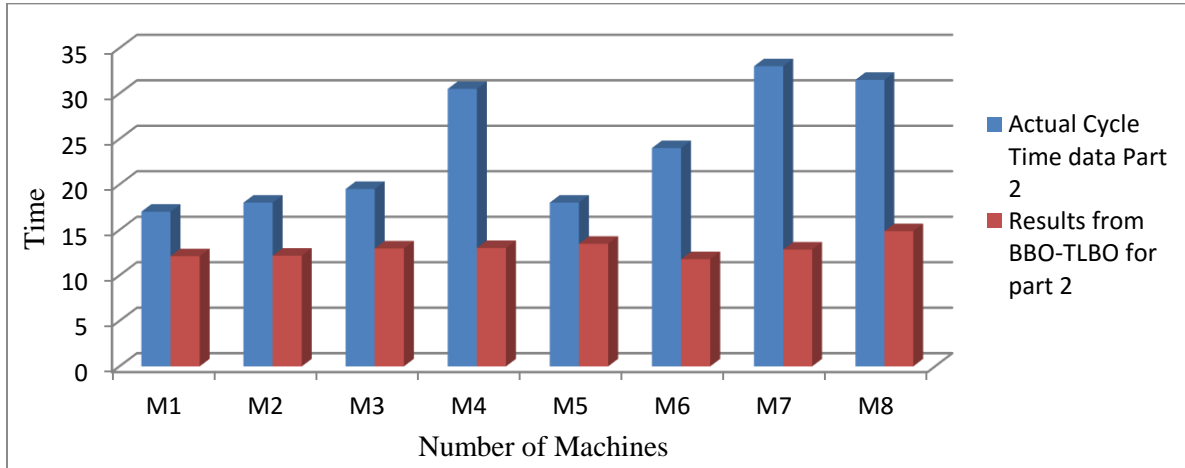


Figure 4.32: Graph Displaying the Actual Data of cycle Time and Results Generated (in minutes) from Proposed BBO-TLBO Heuristic for Part 2

Table 4.36: Depiction of Results Generated (in minutes) from Proposed BBO-TLBO heuristic and the Optimized data for Part 2

	M1	M2	M3	M4	M5	M6	M7	M8
Results from BBO-TLBO for part 2	12.106	12.169	12.955	13.012	13.475	11.786	12.840	14.858
Optimized Data for Part 2	4.894	5.831	6.488	17.025	4.525	12.214	20.16	16.642

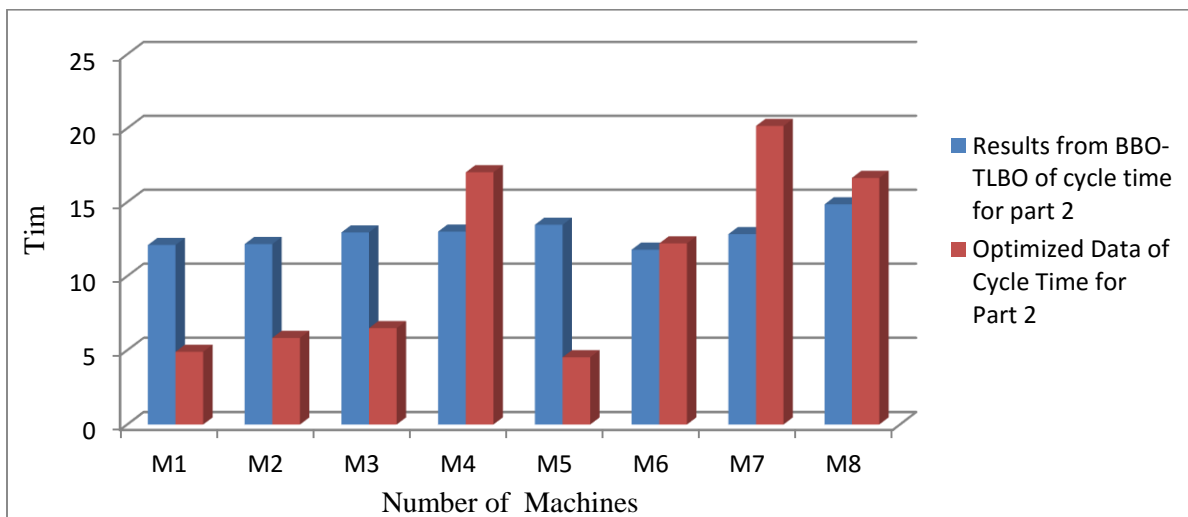


Figure 4.33: Graph showing the Results Generated (in minutes) from Proposed BBO-TLBO Heuristic and the Optimized Data for Part 2

Table 4.36 contains the results obtained from the proposed heuristic and the optimized time. The optimized time is the difference between the actual data and the results obtained. The optimized time shows the quality of the results and the effectiveness of the heuristic in minimizing the values of cycle time for part 2 on all the machines. Figure 4.33 illustrates the results obtained from heuristic and the optimized value of the cycle time for part 1 on all the machines. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The minimum optimized time is 4.894 min on machine M1 and the maximum is 20.16 min on the machine M7. This is evident from the red bars of the graph that heuristic is able to optimize a significantly on all the machines for part 2.

Table 4.37: Depiction of Actual Data and Optimized Data for Part 2

	M1	M2	M3	M4	M5	M6	M7	M8
Actual Cycle Time data Part 2	17	18	19.5	30.5	18	24	33	31.5
Optimized Data for Part 2	4.894	5.831	6.488	17.025	4.525	12.214	20.16	16.642

Table 4.37 shows the actual time spent by the part 2 in the industry and the optimized time obtained from the proposed heuristic. It is clearly evident from the table that the proposed heuristic has been able to optimize the process time for part 2 on all the machines.

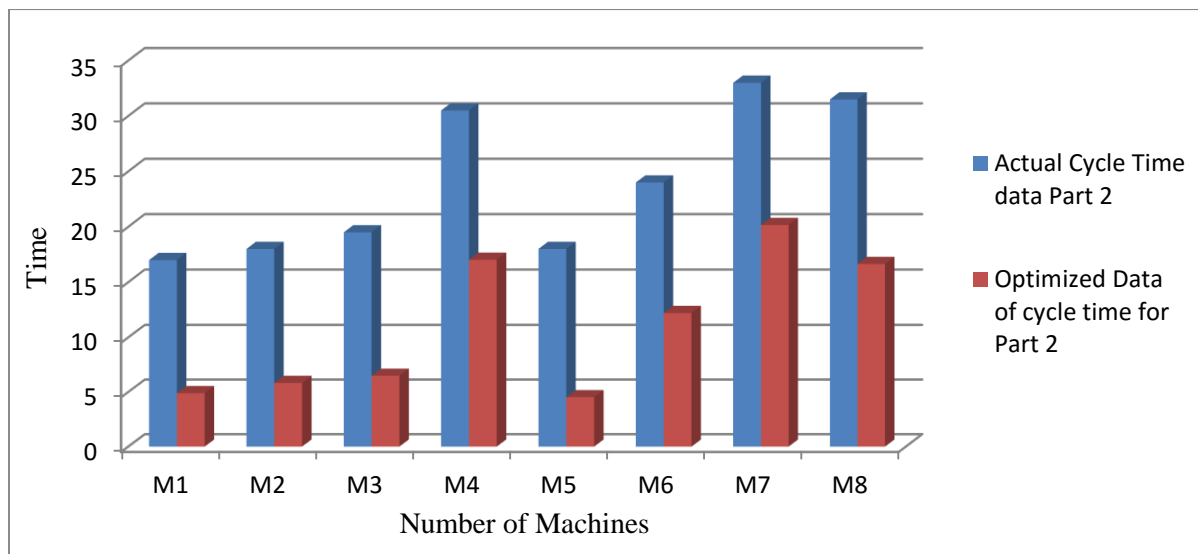


Figure 4.34: Graph Showing the Actual Cycle Time Data and Optimized Cycle Time for Part 2

Figure 4.34 displays the actual cycle time and the optimized time for part 2 on all the machines. The x-axis of the graph displays the number of machines and y-axis displays the processing time. The blue bars shown as the actual time are much bigger than the red bars showing the

optimized time, but are able to significantly cut the slack of the processing time. The most processing time is employed to machine M7, and the most optimized time 20.16 min is on machine M7. Table 4.38 displays the combined set of actual data, results by heuristic BBO-TLBO and the optimized data for part 2 corresponding to each machine. The minimum optimized time is on the machine M5 with 4.525 min and the maximum is on the machine M7 with 20.16 min. The values of the optimized times for all the other machines varies between these two. Figure 4.35 cumulatively displays the actual process time, results from heuristic and the optimized time for part 2. The x-axis of the graph displays the number of machines and y-axis displays the cycle time. The blue bars denote the actual process time on

Table 4.38: Depiction of Actual Data, Results Generated from Proposed BBO-TLBO Heuristic and Optimized Data (in minutes) for Part 2

	M1	M2	M3	M4	M5	M6	M7	M8
Actual Cycle Time data Part 2	17	18	19.5	30.5	18	24	33	31.5
Results from BBO-TLBO for part 2	12.106	12.169	12.955	13.012	13.475	11.786	12.840	14.858
Optimized Data for Part 2	4.894	5.831	6.488	17.025	4.525	12.214	20.16	16.642

the respective machines holds the highest peaks, but the red bars showing the results from heuristic are significant and optimizes the process time on every machine.

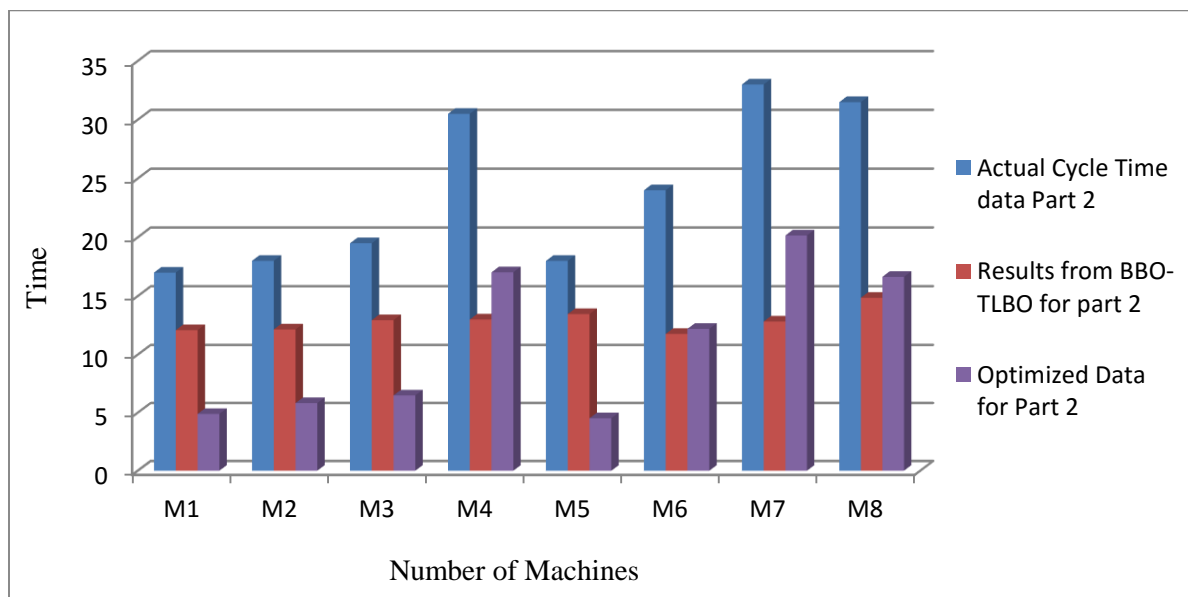


Figure 4.35: Graph Showing the Cumulative Display of Actual Data, Results Generated from Proposed BBO-TLBO Heuristic and Optimized Data (in minutes) for Part 2

4.4.3 Optimization of the Model

Optimization is the process of reducing the function to the minimum value or maximizing the function to its upper limit. The difference between actual value and mean value of the obtained results is considered as the optimized value for the respective part and machine. The optimized result shows the capability of the developed BBO-TLBO heuristic to reduce from a certain minimum to a maximum values between the set of various machines for both the parts, separately. The subsections 4.4.3.1 and 4.4.3.2 described the optimization values for part 1 and part 2 on each machine, respectively.

4.4.3.1 Optimization of Part 1 and Part 2 taking process time as the objective

The optimization of process time takes place on every machine from M1 to M8. But the value of optimized time differs on each machine. Table 4.39 shows the combined process time of part 1 and part 2 on each machine and combined optimized time on that respective machine. On machine M1 the combined process time of part 1 and part 2 is optimized from 23 min by 9.233 min. On machine M2 the combined process time of part 1 and part 2 is optimized from 22 min by 6.209 min. On machine M3 the combined process time of part 1 and part 2 is optimized from 27 min by 10.619 min. On machine M4 the combined process time of part 1 and part 2 is optimized from 40 min by 23.976 min. On machine M5 the combined process time of part 1 and part 2 is optimized from 24 min by 6.117 min. On machine M6 the combined process time of part 1 and part 2 is optimized from 29 min by 11.685 min. On machine M7 the combined process time of part 1 and part 2 is optimized from 39 min by 22.044 min. On machine M8 the combined process time of part 1 and part 2 is optimized from 32 min by 14.364 min.

Table 4.39: Cumulative Depiction of Total Process Time and Total Optimized Time for Part 1 and Part 2 on Various Machines

Part 1 and Part 2	M1	M2	M3	M4	M5	M6	M7	M8
Total Process Time	23	22	27	40	24	29	39	32
Total Optimized Process Time	9.233	6.209	10.619	23.976	6.117	11.685	22.044	14.364

Figure 4.36 depicts the comparison between the total processing time and optimized processing time of part 1 and part 2 by the results obtained from implementation of developed BBO-TLBO

heuristic. The most optimized machine time for part 1 and part 2 is on machine M4 where 23.976 min are optimized from the combined processing time of 40 min. The least optimized time is found on machine M2 where the combined process time of 22 min is optimized by 6.209 min. The rest of the machines are optimized within the range of 6.209 min to 23.976 min.

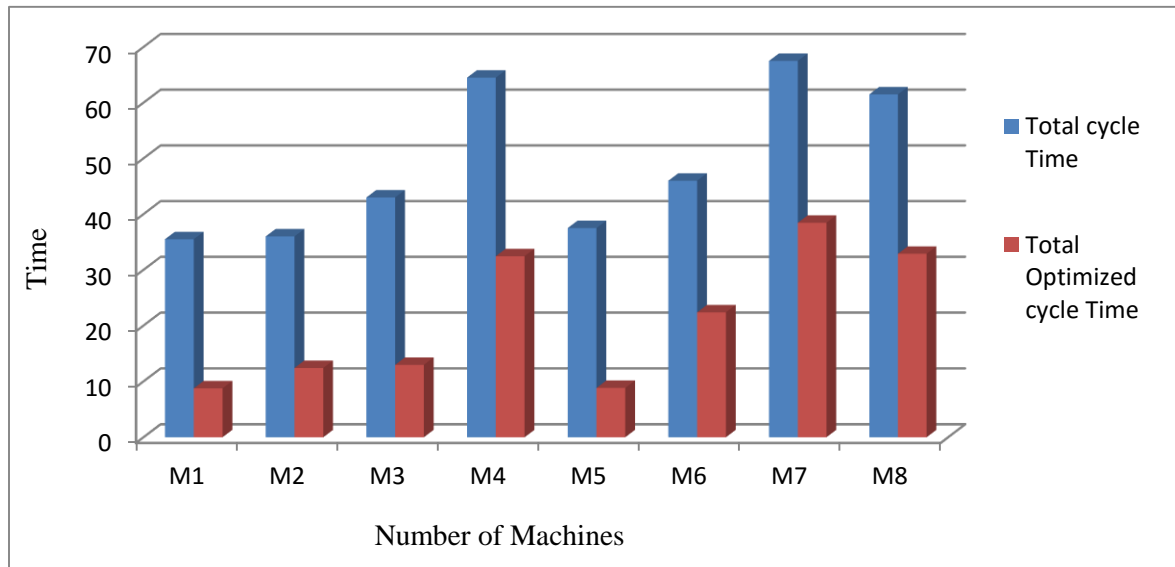


Figure 4.36: Graph Depicting the Total Process Time and Total Optimized Time for Part 1 and Part 2 on Various Machines

4.4.3.2 Optimization of the constraints set up time and backlogging of part 1 and part 2 taking cycle time as the objective

The optimization of cycle time which includes the sequence dependent set up time and backlogging time for both the parts takes place on every machine from M1 to M8. The value of optimized cycle time differs on each machine. Table 4.40 shows the combined process time of part 1 and part 2 on each machine and combined optimized time on that respective machine. On machine M1 the combined process time of part 1 and part 2 is optimized from 35.5 min by 8.755 min. On machine M2 the combined process time of part 1 and part 2 is optimized from 36 min by 12.405 min. On machine M3 the combined process time of part 1 and part 2 is optimized from 43 min by 12.475 min. On machine M4 the combined process time of part 1 and part 2 is optimized from 64.5 min by 32.442 min. On machine M5 the combined process time of part 1 and part 2 is optimized from 37.5 min by 8.838 min. On machine M6 the combined process time of part 1 and part 2 is optimized from 42 min by 22.372 min. On

machine M7 the combined process time of part 1 and part 2 is optimized from 67.5 min by 38.453 min. On machine M8 the combined process time of part 1 and part 2 is optimized from 61 min by 32.886 min.

Table 4.40: Cumulative Depiction of Total Cycle Time and Total Optimized Time for Part 1 and Part 2 on Various Machines

Part 1 and Part 2	M1	M2	M3	M4	M5	M6	M7	M8
Total cycle Time	35.5	36	43	64.5	37.5	46	67.5	61.5
Total Optimized cycle Time	8.755	12.405	12.975	32.442	8.838	22.372	38.453	32.886

Figure 4.37 shows the comparison between the total cycle time and optimized cycle time of part 1 and part 2 by the results obtained from implementation of developed BBO-TLBO heuristic. The most optimized machine time for part 1 and part 2 is on machine M8 where 32.886 min are optimized from the combined processing time of 61.5 min. The least optimized time is found on machine M1 where the combined process time of 35.5 min is optimized by 8.755 min. The rest of the machines are optimized between the range 8.755 min to 32.886 min.

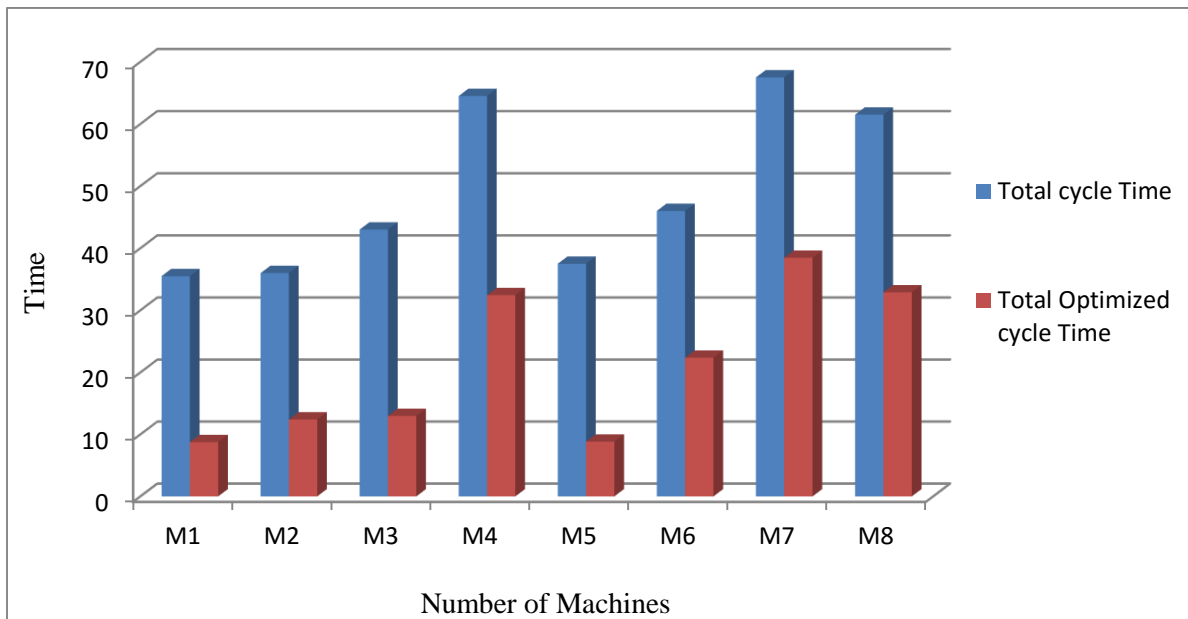


Figure 4.37: Graph Depicting the Total Cycle Time and Total Optimized Cycle Time for Part 1 and Part 2 on Various Machines

4.5 DISCUSSION OF RESULTS

The flow shop scheduling problems are studied from the past many decades by utilizing the many heuristics as described in the literature review. Here, a new heuristic proposed named as BBO-TLBO which utilizes the features of both the methods has been employed. The software used to run the heuristic and evaluate the results was MATLAB. The software was run on Windows 10 1.5 GHz quadcore processor with 4 GB RAM. The processing time for each function was approximately 3 sec. The processing time of the software depends upon the number of iterations and the configuration of the computer system used. More the number of iterations more will be time taken by the processor to evaluate the results. The two objectives which are solved using this software are minimization of the process time and cycle time, respectively. As the two objectives are being solved, so the considered problem can be termed as multi-objective flow shop scheduling. The subsections 4.5.1 and 4.5.2 below discusses the results related to each objective separately.

4.5.1 Discussion of Results for Minimization of Process Time

Table 4.1 to 4.8 and Figure 4.2 to 4.9 show the results produced using BBO-TLBO for the different functions for the minimization of process time for both the parts. The function $11x(1) + 12x(2) \leq 60$ is solved in table 4.1, which reduces the actual value of processing time from 11 min of part 1 to 6.365 min and 12 min of part 2 to 7.402 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 1.552 to 10.964 and for part 2 ranges from 1.249 to 14.865. The minimum and maximum value of function is 53.891 and 248.132, respectively. The function $11x(1) + 11x(2) \leq 90$ is solved in table 4.2, which reduces the value of processing time of part 1 from 11 min to 8.252 min and that of part 2 from 11 min to 7.266 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 1.817 to 13.268 and for part 2 ranges from 1.741 to 13.176. The minimum and maximum value of function is 39.147 and 286.506, respectively. The function $12x(1) + 15x(2) \leq 90$ is solved in table 4.3, which reduces the value of processing time of part 1 from 12 min to 8.816 min and that of part 2 from 15 min to 7.565 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 1.889 to 14.099 and for part 2 ranges from 1.005 to 14.621. The minimum and maximum value of function is 47.082 and 359.482, respectively. The function $19x(1) + 21x(2) \leq 100$ is solved in table 4.4, which reduces the value of processing time of part 1 from 19 min to 8.460 min and that of part 2 from

21 min to 7.564 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 2.577 to 13.019 and for part 2 ranges from 1.325 to 13.288. The minimum and maximum value of function is 106.110 and 508.111, respectively. The function $11x(1) + 13x(2) \leq 60$ is solved in table 4.5, reduces the value of processing time of part 1 from 11 min to 9.572 min and that of part 2 from 13 min to 8.311 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 2.359 to 14.405 and for part 2 ranges from 1.499 to 14.424. The minimum and maximum value of function is 75.049 and 345.971, respectively. The function $15x(1) + 14x(2) \leq 120$ is solved in table 4.6, which reduces the value of processing time of part 1 from 15 to 8.723 and that of part 2 from 14 min to 8.592. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 1.288 to 14.615 and for part 2 ranges from 1.101 to 14.556. The minimum and maximum value of function is 44.559 and 389.070, respectively. The function $19x(1) + 14x(2) \leq 90$ is solved in table 4.7, which reduces the value of processing time of part 1 from 19 min to 7.613 min and that of part 2 from 14 min to 8.343 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 1.371 to 14.924 and for part 2 ranges from 1.897 to 13.463. The minimum and maximum value of function is 178.588 and 556.059, respectively. The function $15x(1) + 17x(2) \leq 120$ is solved in table 4.8, which reduces the value of processing time of part 1 from 15 min to 9.343 min and that of part 2 from 17 min to 8.206 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 1.285 to 14.762 and for part 2 ranges from 1.578 to 14.698. The minimum and maximum value of function is 87.518 and 463.758, respectively.

4.5.2 Discussion of Results for Minimization of Cycle Time

Table 4.9 to 4.16 and Figure 4.10 To 4.17 show the results generated by using BBO-TLBO heuristic for all the functions related cycle time in order to minimize the objective. The function $17x(1) + 18.5x(2) \leq 60$ is solved in table 4.9, which reduces the value of cycle time of part 1 from 17 min to 13.139 min and that of part 2 from 18.5 min to 12.106 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 2.680 to 23.770 and for part 2 ranges from 2.557 to 23.640. The minimum and maximum value of function is 235.570 and 775.506, respectively. The function $18x(1) + 18x(2) \leq 90$ is solved in table 4.10, which reduces the value of cycle time of part 1 from 18 min to 11.426 min and that of part 2 from 18 min to 12.169 min. The value of the variable used by proposed heuristic in the

analysis for part 1 ranges from 1.904 to 23.748 and for part 2 ranges from 4.635 to 21.626. The minimum and maximum value of function is 206.277 and 716.629, respectively. The function $19.5 \times (1) + 23.5 \times (2) \leq 90$ is solved in table 4.11, which reduces the value of cycle time of part 1 from 19.5 min to 13.013 min and that of part 2 from 23.5 min to 12.955 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 1.558 to 23.894 and for part 2 ranges from 1.174 to 23.277. The minimum and maximum value of function is 110.574 and 908.785, respectively. The function $30.5 \times (1) + 34 \times (2) \leq 100$ is solved in table 4.12, which reduces the value of cycle time of part 1 from 30.5 min to 15.083 min and that of part 2 from 34 min to 13.012 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 3.234 to 23.018 and for part 2 ranges from 1.792 to 23.833. The minimum and maximum value of function is 1486.031 and 294.867, respectively. The function $18 \times (1) + 19.5 \times (2) \leq 60$ is solved in table 4.13, which reduces the value of cycle time of part 1 from 18 min to 13.687 min and that of part 2 from 19.5 min to 13.475 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 1.474 to 23.368 and for part 2 ranges from 1.166 to 23.029. The minimum and maximum value of function is 805.804 and 71.733, respectively. The function $24 \times (1) + 22 \times (2) \leq 120$ is solved in table 4.14, which reduces the value of cycle time of part 1 from 24 min to 13.842 min and that of part 2 from 22 min to 11.786 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 2.461 to 22.520 and for part 2 ranges from 1.008 to 23.377. The minimum and maximum value of function is 103.739 and 983.437, respectively. The function $33 \times (1) + 34.5 \times (2) \leq 100$ is solved in table 4.15, which reduces the value of cycle time of part 1 from 33 min to 14.707 min and that of part 2 from 34.5 min to 12.840 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 1.468 to 22.782 and for part 2 ranges from 1.949 to 23.704. The minimum and maximum value of function is 253.799 and 1562.736, respectively. The function $31.5 \times (1) + 30 \times (2) \leq 120$ is solved in table 4.16, which reduces the value of cycle time of part 1 from 31.5 min to 13.251 min and that of part 2 from 30 min to 14.858 min. The value of the variable used by proposed heuristic in the analysis for part 1 ranges from 4.293 to 23.354 and for part 2 ranges from 1.535 to 23.213. The minimum and maximum value of function is 201.215 and 1404.033, respectively.

CHAPTER 5: CONCLUSIONS AND FUTURE SCOPE OF WORK

The scheduling problems have always been the center of research right from the past many decades. Out of the various types of scheduling, flow shop scheduling (FSS) related problems has garnered the attraction of researchers due to its vast utilization in the global industries. The flow shop environment demands the same process sequence of all the products which makes it easy for a manufacturing unit to execute and produce variety of products. Many objectives have been considered by the researchers over the time such as makespan, tardiness, earliness, completion time and so on. The objectives considered in the present research are minimization of process time and cycle time, respectively. Due to the two undertaking of two objectives, the present proposed problem becomes multi-objective flow shop scheduling problem (MFSP). Sequence Dependent Setup Times (SDST) and introduction of Backlogging in the cycle time makes the problem more complex. Efficient handling of Sequence Dependent Set-up Times (SDST) is one of the important elements to increase manufacturing organization performance and experiences to be looked separately from the processing time. This type of MFSP problems are considered as NP hard, whose exact solutions cannot be determined. The various heuristics developed and used in the previous researches are discussed in Chapter 1 of introduction and Chapter 2 of literature models. The present proposed work employs two young heuristics namely, Biogeography Based Optimization (BBO) and Teacher Learning Based Optimization (TLBO). These both are combined together in order to design a novel hybrid heuristic utilizing the migration and mutation phase of BBO, along with teacher phase of TLBO. TLBO's teacher phase based BBO heuristic examines stated scheduling problem. The proposed heuristic is able to minimize the process time and cycle time of two parts, machining on the eight different machines. Further, the comparisons have been shown in order to validate the results from the proposed heuristic corresponding to the actual data used to form the objective functions. The analytical results address that the proposed heuristic is effective in minimizing the desired functions and fully finds its application in flow shop scheduling environment.

5.1 CONCLUSIONS

In the present research, a new heuristic model is proposed depending on search algorithms for production scheduling and also developed a programming in MATLAB to solve multi-objective scheduling problems to develop the present scheduling process efficient. The

proposed algorithm can be used for analyzing scheduling problems up to eight machines. Otherwise, there are some cases to be considered as mentioned. Some of the major discovering conclusions from the confront work experienced as follows:

(i) The process time and the cycle time have been minimized using the proposed BBO-TLBO heuristic and the optimized results contribute that the variation in the objective function. The respective function will goes on decreasing with increase in the number of iterations. The successive iterations utilize the different value of the variables, hence maintaining the diversity in the set of solutions.

(ii) Sequence Dependent Setup Time (SDST) and Backlogging time were configured with multi-objective flow shop scheduling problem to make the system more realistic. This is accomplished as multiple decisions are often required in this dynamic and competitive surrounding. Keeping these conflicting conditions in mind, the propose heuristic tends to find the effective solutions undertaking then concerned conditions.

(iii) The actual data i.e., process time and cycle time are optimized using proposed BBO-TLBO heuristic. For part 1, the processing time on the eight machines has been optimized ranging between 1.4 to 11.3 minutes and optimized process time of part 2 ranges between 3.7 to 13.4 minutes. The optimized cycle time of part 1 ranges between 3.8 to 10.3 minutes and optimized cycle time of part 2 ranges between 4.8 to 12.2 minutes. The population size is 20, number of variables are 2, number of iterations are 20. Therefore, the reduced time refers the effectiveness of the proposed hybrid heuristic algorithm and hence, it makes the system more reliable.

(iv) The Performance of proposed hybrid BBO-TLBO heuristic displays the wide range of results to choose from and analyze the functions deeply. This is evident from the examination of the results which shows the tendency of heuristic to achieve the bounded values. Hence, it is capable of formulating most beneficial results. The optimized results will be affected with changes in the number of machines, variables, lower and upper bound limits.

(v) The proposed research will suggest the minimization of other objectives such as earliness, tardiness, total transportation time, delay time, weighted functions as well can be minimized using proposed novel hybrid BBO-TLBO heuristic and the results can be compared with other heuristics algorithms listed in the literature.

(vi) The manufacturing system with many products (jobs) and machines can be studied containing many number of variables as well, when configured with the righteous objective

functions. The formulation of the objective functions needs to be performed separately in order to avoid any miscalculations.

5.2 RECOMMENDATIONS

The following are the recommendations which will be helpful in exploring the present work in order to experiment to make it more reliable.

(i) The present proposed work strongly would recommend to form as much as novel combinations of the selected heuristics while designing or developing the hybrid method. The present study utilizes hybrid BBO-TLBO heuristic to optimize the dual objectives of process time and cycle time. The bucket of combinations will give the researcher an opportunity to decide the best combination for producing results. Moreover, the researcher can be capable of designing a thoroughly new method for the concerned problem.

(ii) The proposed research work would suggest the selection of the objectives, parameters and their values should be in accordance to the standards in the base paper, industrial reports from authenticated and peer-reviewed before formulation of the objective functions. The objective functions in the present work are framed after analyzing the actual data from the industry. This ensures the quality of data and moreover, the results can be reverted back to the industry to make the same manufacturing unit more efficient than earlier system.

(iii) The present study would recommend the selection of various values of constants such as lower bound; upper bound; number of variables; population rates and number of iterations and so on which remain constant throughout the iterating process based upon the requirement of data to be optimized and the feasibility of post results. The quality of solutions generated will be affected by the variation in the values of the former. Further, it is also recommended, if possible, to choose different values of the above constants according to the objective function under consideration.

(iv) The present work would suggest to include the various real life constraints as investigated in the present work, such as SDST and backloging, so as to apply the results obtained by utilizing the developed method in real life scenario as well. The constraints are necessary because of the complex multi-product manufacturing units used in the global market to produce a wide range of products on the same unit and it is often that different products are manufactured with different constraints as they differ in specifications.

(v) The proposed study would synthesize application of MATLAB for the evaluation of optimization of linear and non-linear functions. In the proposed work, objective functions for minimization of process time and cycle time are successfully evaluated using BBO-TLBO hybrid heuristic coded in MATLAB. The research work recommends the subjective study of software beforehand in order to obtain familiarity of modelling in the software. Further, the present work recommends utilization of contemporary software such as C, C++, C #, JAVA and others, as an alternative for the evaluation of the objective functions. These all differ in coding but capable of delivering the results of iterative loops and are highly utilized by researchers, as reviewed in literature.

5.3 FUTURE SCOPE OF WORK

The proposed research work presented a design of novel hybrid heuristic for multi-objective flow shop scheduling. Hence it is very naive and open to the any type of explorations which expands the horizon of the proposed BBO-TLBO heuristic. Below are some of the aspects in which the present work can be made more versatile for scheduling problems.

(i) The proposed work has been restricted to define the performance measures like minimizing process time and cycle time in the proposed BBO-TLBO heuristic. There are number of objectives such as total weighted squared tardiness, earliness, make-span, total weighted squared earliness and number of tardy jobs, delay time, completion time and so on which can be solved using the proposed hybrid heuristic. These objectives can be solved individually or taken multiple at time.

(ii) Advancements in creating a user friendly interaction in MATLAB to enable the user to easily input the values and moreover to modify any functions, will prove to be a significant improvement. Not only, the modifications if the software may be able to show some relevant information regarding the respective methods used, will be a revolutionary step in the working experience of the software world. Moreover, some other software such as JAVA, C, C++ can be utilized to code the same heuristics.

(iii) The proposed heuristics can be combined with other heuristics, enlisted in the literature such as GA, SA, PSO, TS, IA, DE or any other recently developed methods to make it more competent with the contemporary heuristics used to solve scheduling problems. Also, the

proposed BBO-TLBO can be used to solve problems of fields other than mechanical, such as computer, electronics, and food processing and so on.

(iv) The proposed heuristics can be developed further to give the faster results with the increased number of iterations so as to make the heuristic more efficient. The quality of the solutions can also be seen as an aspect to focus on, by making some amendments in the parameters in order to generate better results. The effectiveness of algorithms is always under radar, and analyzing the results delivered in present work, selection of the variables to obtain the function value can be a subject of improvement during the working of heuristic.

(v) The working of proposed heuristic can be made dynamic so as to enable the user to input or modify the values in between the process. Dynamicity of algorithms has always remained a big task among the researchers. The current world manufacturing systems demands to be more competent, efficient with high production volumes of the diversified specifications of the products. Hence, developing the BBO-TLBO heuristic to attain dynamicity will nurture novel and reliable results for the industries to make manufacturing systems more effective.

REFERENCES

1. Abdelsadek, Y., Herrmann, F., Kacem, I. and Otjacques, B. (2015), “*Branch-and-Bound Algorithm for the Maximum Triangle Packing Problem*”, *Computers & Industrial Engineering*, Vol. 81, pp. 147-157.
2. Abido, M. A. (2002), “*Optimal Power Flow Using Particle Swarm Optimization*”, *International Journal of Electrical Power & Energy Systems*, Vol. 24, 7, pp. 563-571.
3. Ahmadizar, F. and Farahani, M. H. (2012), “*A Novel Hybrid Genetic Algorithm for the Open-Shop Scheduling Problem*”, *International Journal of Advanced Manufacturing Technology*, Vol. 62, pp. 775–787.
4. Al-Anzi, F. S. and Allahverdi, A. (2007), “*A Self-Adaptive Differential .Evolution Heuristic for Two-Stage Assembly Scheduling Problem to Minimize Maximum Lateness with Setup Times*” , *European Journal of Operational Research*, Vol. 182, 1, pp. 80-94.
5. Alexouda, G. and Paparrizos, K. (2001), “*A Genetic Algorithm Approach to the Product Line Design Problem using the Seller's Return Criterion: An Extensive Comparative Computational Study*”, *European Journal of Operational Research*, Vol. 134, 1, pp. 165-178.
6. Altinoz, O. T. and Yilmaz, A. E.(2012) “*Particle Swarm Optimization with Parameter Dependency Walls and its Sample Application to the Microstrip-like Interconnect Line Design*”, *AEU - International Journal of Electronics and Communications*, Vol. 66, 2, pp. 107-114.
7. Amini, A., Tavakkoli-Moghaddam, R. (2016), “*A Bi-Objective Truck Scheduling Problem in a Crossdocking Center with Probability of Breakdown for Trucks*”, *Computers & Industrial Engineering*, DOI: 10.1016/j.cie.2016.03.023.
8. Armentano, V. A. and Ronconi, D. P. (1999), “*Tabu Search for Total Tardiness Minimization in Flow-shop Scheduling Problems*” , *Computers & Operations Research*, Vol. 26, 3, pp. 219-235.
9. Azadeh, A., Shoja, B.M., Ghanei, S. and Sheikhalishahi, M. (2015), “*A Multi-Objective Optimization Problem for Multi-State Series-Parallel Systems: A Two-Stage Flow-Shop Manufacturing System*”, *Reliability Engineering & System Safety*, Vol. 136, pp. 62-74.
10. Azadeh, A., Taghipour, M., Asadzadeh, S.M. and Abdollahi, M. (2014), “*Artificial Immune Simulation for Improved Forecasting of Electricity Consumption with Random*

- Variations”, International Journal of Electrical Power & Energy Systems, Vol. 55, pp. 205-224.
11. Azzi, A., Faccio, M., Persona, A. and Sgarbossa, F. (2012), “*Lot Splitting Scheduling Procedure for Make-span Reduction and Machine Capacity Increase in A Hybrid Flow Shop with Batch Production*” International Journal of Advanced Manufacturing Technology, Vol. 59, pp. 775–786.
 12. Babaei M., Mohammadi, M. and Ghomi, S. M. T. F. (2014), “*A Genetic Algorithm for the Simultaneous Lot Sizing and Scheduling Problem in Capacitated Flow Shop with Complex Setups and Backlogging.*” International Journal of Advanced Manufacturing Technology, Vol. 70, pp. 125–134.
 13. Babaei, M., Mohammadi, M. and Ghomi, S. M. T. F. (2014), “*A Genetic Algorithm for the Simultaneous Lot Sizing and Scheduling Problem in Capacitated Flow Shop with Complex Setups and Backlogging*”, International Journal of Advanced Manufacturing Technology, Vol. 70, pp. 125–134.
 14. Babu, B.V. and Shastry, K. K. N. (1999) “*Estimation of Heat Transfer Parameters in a Trickle-Bed Reactor Using Differential Evolution and Orthogonal Collocation*”, Computers & Chemical Engineering, Vol. 23, 3, pp. 327-339.
 15. Batur, G.D., Erol, S. and Karasan, O. E. (2016), “*Robot Move Sequence Determining and Multiple Part-Type Scheduling in Hybrid Flexible Flow Shop Robotic Cells*”, Computers & Industrial Engineering, Vol. 100, pp. 72–87.
 16. Baykasoglu, A., Hamzadayi, A. and Kose, S.Y. (2014), “*Testing the Performance of Teaching–Learning Based Optimization (TLBO) Algorithm on Combinatorial Problems: Flow shop and Job Shop Scheduling Cases*”, Information Sciences.
 17. Bellman, R., Esogbue, A. O. and Nabeshima I. (1982), “*Mathematical Aspects of Scheduling and Applications*”, Pergamon Press, Oxford, pp. 258-280.
 18. Ben-Daya, M. and Al-Fawzan, M. (1998), “*A Tabu Search Approach for the Flow Shop Scheduling problem*”, European Journal of Operational Research, Vol. 109, 1, pp. 88-95.
 19. Bera, S. and Mukherjee, I. (2016), “*A Multistage and Multiple Response Optimization Approach for Serial Manufacturing System*”, European Journal of Operational Research, Vol. 248, 2, pp. 444-452.

20. Berthiau, G. and Siarry, P. (1993), “An *Enhanced Simulated Annealing Algorithm for Extracting Electronic Component Model Parameters*”, *Advances in Engineering Software*, Vol. 18, 3, pp. 171-176.
21. Bessedik, M., Tayeb, F. B., Cheurfi, H. and Blizak, A. (2015), “An *Immunity-Based Hybrid Genetic Algorithms for Permutation Flow-shop Scheduling Problems*”, *International Journal of Advanced Manufacturing and Technology*, Vol. 85, 9, pp. 2459-2469.
22. Bhattacharya, A. and Chattopadhyay, P.K. (2010), “Solving Complex Economic Load Dispatch Problems Using Biogeography-Based Optimization”, *Expert Systems with Applications*, Vol. 37, 5, pp. 3605–3615.
23. Bilde, O. and Krarup, J. (1977), “*Sharp Lower Bounds and Efficient Algorithms for the Simple Plant Location Problem*”, In: P.L. Hammer, E.L. Johnson, B.H. Korte and G.L. Nemhauser, Editor(s), *Annals of Discrete Mathematics*, Elsevier, Vol. 1, pp. 79-97.
24. Botta-Genoulaz, V. (2000), “*Hybrid Flow Shop Scheduling with Precedence Constraints and Time Lags to Minimize Maximum Lateness*”, *International Journal of Production Economics*, Vol. 64, 1–3, pp. 101-111.
25. Boxma, O. J. and Forst, F. G. (1986), “*Minimizing the Expected Weighted Number of Tardy Jobs in Stochastic Flow Shops*”, *Operations Research Letters*, Vol. 5, 3, pp. 119-126.
26. Brito, A. J. and de Almeida, A.T. (2012), “*Modeling A Multi-Attribute Utility Newsvendor With Partial Backlogging*”, *European Journal of Operational Research*, Vol. 220, 3, pp. 820-830.
27. Chao, I.M. (2002), “*A Tabu Search Method for the Truck and Trailer Routing Problem*”, *Computers & Operations Research*, Vol. 29, Issue 1, pp 33-51.
28. Chardaire, P., Lutton, J. L. and Sutter, A. (1993), “*Thermo statistical Persistency: A Powerful Improving Concept for Simulated Annealing Algorithms*”, *European Journal of Operational Research*, Vol. 86, 3, pp. 565-579.
29. Chastang, Cl. and Gremy, F. (1978), “*A Computer Program for Parenthood Diagnosis within a Family*”, *Computer Programs in Biomedicine*, Vol. 8, 1, pp. 1-15.
30. Chen, D., Zou, F., Wang, J. and Li, S. (2015), “*An Improved Teaching–Learning-Based Optimization Algorithm for Solving Global Optimization Problem*”, *Information Sciences*, Vol. 297, pp. 171–190.

31. Christofides, Alvarez-Valdes, R. and Tamarit, J. M. (1987), “*Project Scheduling with Resource Constraints: A Branch and Bound Approach*”, European Journal of Operational Research, Vol. 29, 3, pp. 262-273.
32. Chutima, P. and Naruemitwong, W. (2014), “*A Pareto Biogeography-Based Optimisation for Multi-Objective Two-Sided Assembly Line Sequencing Problems with a Learning Effect*”, Computers & Industrial Engineering Vol.69, pp. 89–104.
33. Ciavotta, M., Minella, G. and Ruiz, R. (2013), “*Multi-Objective Sequence Dependent Setup Times Permutation Flow-shop: A New Algorithm and a Comprehensive Study*”, European Journal of Operational Research, Vol. 227, 2, pp. 301-313.
34. Costa, A., Alfieri, A., Matta, A. and Fichera, S. (2015), “*A Parallel Tabu Search for Solving the Primal Buffer Allocation Problem in Serial Production Systems*”, Computers & Operations Research, Vol. 64, pp. 97-112.
35. Costa, A., Cappadonna, F. A. and Fichera, S. (2014), “*A Novel Genetic Algorithm for the Hybrid Flow Shop Scheduling with Parallel Batching and Eligibility Constraints*”, International Journal Advance Manufacturing Technology, Vol. 75, pp. 833–847.
36. Costa, A., Cappadonna, F.A. and Fichera, S. (2012), “*A Novel Genetic Algorithm for the Hybrid Flow Shop Scheduling with Parallel Batching and Eligibility Constraints.*”, International Journal Advanced Manufacturing Technology, Vol. 75, pp. 833–847.
37. Cruz, I.L.L., Willigenburg, L.G.V. and Straten, G.V. (2005), “*Efficient Differential Evolution Algorithms for Multimodal Optimal Control Problems*”, Applied Soft Computing, Vol. 3, Issue 2, pp. 97-122.
38. Czapinski, M. (2010), “*Parallel Simulated Annealing with Genetic Enhancement for Flow-shop Problem with Csum*”, Computers & Industrial Engineering, Vol. 59, 4, pp. 778-785.
39. Damak, N., Jarboui, B., Siarry, P. and Loukil, T. (2009), “*Differential Evolution for Solving Multi-Mode Resource-Constrained Project Scheduling Problems*”, Computers & Operations Research, Vol. 36, 9, pp. 2653-2659.
40. Dolan, W. B., Cummings, P. T. and Le Van, M. D. (1990), “*Algorithmic Efficiency of Simulated Annealing for Heat Exchanger Network Design*”, Computers & Chemical Engineering, Vol.14, 10, pp. 1039-1050.

41. Dong, M. G. and Wang, N. (2012), “*A Novel Hybrid Differential Evolution Approach to Scheduling of Large-Scale Zero-Wait Batch Processes with Setup Times*”, *Computers & Chemical Engineering*, Vol. 45, pp. 72-83.
42. Dorigo M. and Gambardella, L. M. (1997), “*Ant Colonies for the Travelling Salesman Problem*”, *Biosystems*, Vol. 43, 2, pp. 73-81.
43. Dorigo, M. and Gambardella, L.M. (1997), “*Ant Colony System, “A Cooperative Learning Approach to the Traveling Salesman Problem*”, *IEEE Transactions on Evolutionary Computation*, Vol 1, 1, pp. 53-66.
44. Elyasi, A. and Salmasi, N. (2013), “*Stochastic Flow-Shop Scheduling with Minimizing The Expected Number of Tardy Jobs*”, *International Journal of Advanced Manufacturing Technology*, Vol. 66, pp. 337–346.
45. Farmer, J. D., Packard, N. and Perelson, A. (1986), “*The Immune System, Adaptation and Machine Learning*”, *Physica D*, Vol. 2, pp. 187–204.
46. Fernandez-Viagas, V. and Framinan, J.M. (2015), “*Efficient Non-Population-Based Algorithms for the Permutation Flow-Shop Scheduling Problem with Make-span Minimization Subject to a Maximum Tardiness*”, *Computers & Operations Research*, Vol. 65, pp. 86–96.
47. Flynn, B. B. (1987), “*Repetitive lots: The use of a Sequence-Dependent Set-up Time Scheduling Procedure in Group Technology and Traditional Shops*”, *Journal of Operations Management*, Vol. 7, 1, pp. 203-216.
48. Freire, A. S., Moreno, E. and Yushimito, W. F. (2016), “*A Branch-and-Bound Algorithm for the Maximum Capture Problem with Random Utilities*”, *European Journal of Operational Research*, Vol. 252, 1, pp. 204-212.
49. Gajpal, Y. and Abad, P. L. (2009), “*Multi-Ant Colony System (MACS) for a Vehicle Routing Problem with Backhauls*”, *European Journal of Operational Research*, Vol. 196, 1, pp. 102-117.
50. Gedik, R., Rainwater, C., Nachtmann, H. and Pohl, E. A. (2015), “*Analysis of a Parallel Machine Scheduling Problem with Sequence Dependent Setup Times and Job Availability Intervals.*” *European Journal of Operational Research*, DOI: 10.1016/j.ejor.2015.11.020

51. Gerstl, E. and Mosheiov, G. (2014), “*A Two-Stage Flexible Flow Shop Problem with Unit-Execution-Time Jobs and Batching*”, *International Journal Production Economics*, Vol. 158, pp. 171–178.
52. Glover, F. (1986), “*Future Paths for Integer Programming and Links to Artificial Intelligence*”, *Computers and Operations Research*, Vol. 13, 5, pp. 533–549.
53. Glover, F. (1989), “*Tabu Search-Part I*”, *ORSA Journal on Computing*, Vol. 1, 2, pp. 190–206.
54. Gourgand, M., Grangeon, N. and Norre, S. (2003), “*A Contribution to the Stochastic Flow Shop Scheduling Problem*”, *European Journal of Operational Research*, Vol. 151, 2, pp. 415-433.
55. Grabowski, J. and Wodecki, M. (2004), “*A Very Fast Tabu Search Algorithm for the Permutation Flow Shop Problem with Make-span Criterion*”, *Computers & Operations Research*, Volume 31, 11, pp. 1891-1909.
56. Graham R. L., Lawler E. L., Lenstra, J. K. and Rinnooy, K. A. H. G (1979),”*Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey*”, *Annals of Discrete Mathematics*, Volume 5, pp. 287-326.
57. Graves, S.C., Meal, H.C., Stefek, D. and Zeghmi, A. K. (1983), “*Scheduling of Re-Entrant Flow Shops*”, *Journal of Operations Management*, Vol. 3, 4, pp. 197-207.
58. Guo, Z.X., Wang, W., Liu, L., Yang, J. (2015), “*Harmony Search-Based Multi-Objective Optimization Model for Multi-Site Order Planning with Multiple Uncertainties and Learning Effects*”, *Computers & Industrial Engineering*, Vol. 12, 1, pp. 102-119.
59. Halim, A.H., Miyazaki, S. and Ohta, H. (1994), “*Batch-Scheduling Problems to Minimize Actual Flow Times of Parts through the Shop under JIT Environment*”, *European Journal of Operational Research*, Vol. 72, 3, pp. 529-544.
60. Han, Y., Gong, D., Jin, Y. and Pan, Q., (2016), “*Evolutionary Multi-Objective Blocking Lot-Streaming Flow Shop Scheduling With Interval Processing Time*”, *Applied Soft Computing*, Vol. 42, pp. 229-245.
61. Hani, Y., Amodeo, L., Yalaoui, F. and Chen, H. (2007), “*Ant Colony Optimization for Solving an Industrial Layout Problem*”, *European Journal of Operational Research*, Volume 183, 2, 1, pp. 633-642.

62. Hariri, A. M. A. and Potts, C. N. (1984), "Algorithms for Two-Machine Flow-Shop Sequencing with Precedence Constraints", *European Journal of Operational Research*, Vol. 17, 2, pp. 238-248.
63. Hatami, S., Ruiz, R. and Andrés-Romano, C. (2015), "Heuristics and Meta-heuristics for the Distributed Assembly Permutation Flow-shop Scheduling Problem with Sequence Dependent Setup Times", *International Journal of Production Economics*, Vol. 169, pp. 76–88.
64. Hendy, M. D., and Penny, D. (1982), "Branch and Bound Algorithms to Determine Minimal Evolutionary Trees", *Mathematical Biosciences*, Vol. 59, 2, pp. 277-290.
- Henley, E. J. and Williams, R.A. (1973), "Branch and Bound, Search Tree Methods, *Mathematics in Science and Engineering*", Elsevier, Vol. 98, pp. 189-222.
65. Hertz, A. (1991), "Tabu Search for Large Scale Timetabling Problems", *European Journal of Operational Research*, Vol. 54, 1, pp. 39-47.
66. Hsieh, Y. -C., You, P. -S. and Liou, C. D. (2009), "A Note of Using Effective Immune Based Approach for the Flow Shop Scheduling with Buffers", *Applied Mathematics and Computation*, Vol. 215, 5, pp. 1984-1989.
67. Huang, R. H., Yu, S. C. and Kuo, C. W. (2014), "Reentrant Two-Stage Multiprocessor Flow Shop Scheduling with Due Windows", *International Journal of Advanced Manufacturing Technology*, Vol. 71, pp. 1263–1276.
68. Huang, R., Yu, S. and Kuo, C. (2014), "Reentrant Two-Stage Multiprocessor Flow Shop Scheduling with Due Windows." *International Journal Advanced Manufacturing Technology*, Vol. 71, pp. 1263–1276.
69. Huang, S. J. (1999), "Enhancement of Thermal Unit Commitment Using Immune Algorithms Based Optimization Approaches", *International Journal of Electrical Power & Energy Systems*, Vol. 21, 4, pp. 245-252.
70. Hunsucker, J. L and Shah, J. R. (1992), "Performance of Priority Rules in a Due Date Flow Shop", *Omega*, Vol. 20, Issue 1, pp. 73-89.
71. Sarin, S and Lefoka, M. (1993), "Scheduling Heuristic for the n-job m-machine Flow Shop", *Omega*, Vol. 21, 2, pp. 229-234.

72. Icmeli, O. and Erenguc, S. S. (1994), “A *Tabu Search Procedure for the Resource Constrained Project Scheduling Problem with Discounted Cash Flows*” , Computers & Operations Research, Vol. 21, 8, pp. 841-853.
73. Jain, A. S. and Meeran, S. (2002), “A *Multi-Level Hybrid Framework Applied to the General Flow-Shop Scheduling Problem*”, Computers & Operations Research, Vol. 29, 13, pp. 1873-1901.
74. Javadian, N., Fattahi, P., Farahmand-Mehr, M., Amiri-Aref, M. and Kazemi, M. (2012), “An *Immune Algorithm for Hybrid Flow Shop Scheduling Problem with Time Lags and Sequence-Dependent Setup Times*”, International Journal of Advanced Manufacturing Technology.
75. Jayaswal, S. and Agarwal, P. (2014) “*Balancing U-shaped Assembly Lines with Resource Dependent Task Times: A Simulated Annealing Approach*”, Journal of Manufacturing Systems, Vol. 33, 4, pp. 522-534.
76. Jeong, B and Kim, Y. (2014), “*Minimizing Total Tardiness In A Two-Machine Re-Entrant Flow-shop With Sequence-Dependent Setup Times*”, Computers & Operations Research, Vol. 47, pp. 72-80.
77. Jeong, B and Kim, Y. (2014), “*Minimizing Total Tardiness In A Two-Machine Re-Entrant Flow-shop With Sequence-Dependent Setup Times*”, Computers & Operations Research, Vol. 47, pp. 72-80.
78. Johnson, S. M. (1954), “*Optimal Two and Three Stage Production Schedules with Setup Times Included*”, Naval Research Logistics, Vol.1, 1, pp. 61–68.
79. Kahraman, C., Engin, O., Kaya, I. and Ozturk, R. E. (2010), “*Multiprocessor Task Scheduling in Multistage Hybrid Flow-Shops: A Parallel Greedy Algorithm Approach*”, Applied Soft Computing, Vol. 10, 4, pp. 1293-1300.
80. Karimi, N. and Dvoudpour, H. (2014), “*A High Performing Meta-heuristic for Multi-Objective Flow-Shop Scheduling Problem*”, Computers & Operations Research.
81. Kawtummachai, R., Yanagawa, Y., Ohashi, K. and Miyazaki, S. (1997), “*Scheduling in an Automated Flow Shop to Minimize Cost: Backward-Meta Scheduling Method*”, International Journal of Production Economics, Vol. 49, 3, pp. 225-235.

82. Kellegoz, T. and Toklu, B. (2012), “*An Efficient Branch and Bound Algorithm for Assembly Line Balancing Problems with Parallel Multi-Manned Workstations*”, *Computers & Operations Research*, Vol. 39, 12, pp. 3344-3360.
83. Kennedy, J. and Eberhart, R. (1995), "*Particle Swarm Optimization*". Proceedings of IEEE International Conference on Neural Networks, pp. 1942–1948.
84. Khachaturyan, A., Semenovskaya, S. and Vainshtein, B. (1979), “*Statistical-Thermodynamic Approach to Determination of Structure Amplitude Phases*”, *Soviet Physics, Crystallography*, Vol. 24, 5, pp. 519–524.
85. Khorasani, D. and Moslehi, G. (2017), “*Two-Machine Flow Shop Scheduling Problem with Blocking, Multi-Task Flexibility of the First Machine, and Preemption*”, *Computers and Operation Research*, DOI: 10.1016/j.cor.2016.09.023
86. Kim, Y., Lim, H. and Park, M. (1996), “*Search Heuristics for a Flow-shop Scheduling Problem in a Printed Circuit Board Assembly Process*”, *European Journal of Operational Research*, Vol. 91, 1, pp. 124-143.
87. Kirlik, G. and Oguz, C. (2012), “*A Variable Neighborhood Search For Minimizing Total Weighted Tardiness With Sequence Dependent Setup Times On A Single Machine*”, *Computers & Operations Research*, Vol. 39, 7, pp. 1506-1520.
88. Kumar, A., Prakash, A., Shankar, R. and Tiwari, M. K. (2006), “*Psycho-Clonal Algorithm Based Approach to Solve Continuous Flow Shop Scheduling Problem*”, *Expert Systems with Applications*, Vol. 31, 3, pp. 504-514.
89. Kumar, R. S., Kondapaneni, K, Dixit, V. ,Goswami, A. ,Thakur, L. S. and Tiwari, M. K. (2016), “*Multi-Objective Modeling of Production and Pollution Routing Problem with Time Window: A Self-Learning Particle Swarm Optimization Approach*”, *Computers & Industrial Engineering* , Vol. 99, pp. 29-40.
90. Kyprianou, A., Worden, K. and Panet, M. (2001), “*Identification of Hysteretic Systems Using the Differential Evolution Algorithm*”, *Journal of Sound and Vibration*, Vol. 248, 2, pp. 289-314.
91. Lamghari, A. and Dimitrakopoulos, R. (2012), “*A Diversified Tabu Search Approach for the Open-Pit Mine Production Scheduling Problem with Metal Uncertainty*”, *European Journal of Operational Research*, Vol. 222, 3, pp. 642-652.

92. Li, K. Y. and Willis, R. J. (1992) , “*An Iterative Scheduling Technique for Resource-Constrained Project Scheduling*”, Vol. 56, 3, pp. 370-379.
93. Li, Z., Chen, H., Xu, R. and Li, X. (2015), “ *Earliness–Tardiness Minimization on Scheduling a Batch Processing Machine with Non-identical Job Sizes*”, Computers & Industrial Engineering, Vol. 87, pp. 590–599.
94. Liao, C., Lee, C. and Lee, H. (2015), “*An Efficient Heuristic for a Two-Stage Assembly Scheduling Problem with Batch Setup Times to Minimize Make-span*”, Computers & Industrial Engineering, Vol. 88, pp. 317–325.
95. Liao, T., Stutzle, T., Montes de Oca, M. A. and Dorigo, M. (2014), “A unified ant colony optimization algorithm for continuous optimization”, European Journal of Operational Research, Vol. 234, 3, pp. 597-609.
96. Lim, J. (1997), “*A Genetic Algorithm for a Single Hoist Scheduling in the Printed-Circuit-Board Electroplating Line*”, Computers & Industrial Engineering, Vol. 33, 3, pp. 789-792.
97. Lin, J. and Zhang, S. (2016) “*An Effective Hybrid Biogeography-Based Optimization Algorithm for the Distributed Assembly Permutation Flow-Shop Scheduling Problem.*” Computers & Industrial Engineering, Vol. 97, pp. 128–136
98. Lin, J. and Zhang, S. (2016), “*An Effective Hybrid Biogeography-Based Optimization Algorithm for the Distributed Assembly Permutation Flow-Shop Scheduling Problem*”, Computers & Industrial Engineering, Vol. 97, pp. 128–136.
99. Lin, Q., Chen, J. (2013) “*A Novel Micro-Population Immune Multi-objective Optimization Algorithm*”, Computers & Operations Research, Vol. 40, 6, pp. 1590-1601.
100. Lin, Q., Gao, L., Li, X. and Zhang, C. (2015), “*A Hybrid Backtracking Search Algorithm for Permutation Flow-shop Scheduling Problem*”, Computers & Industrial Engineering, DOI: 10.1016/j.cie.2015.04.009.
101. Lin, Q., Li, J., Du, Z., Chen, J. and Ming, Z. (2015) “*A Novel Multi-Objective Particle Swarm Optimization with Multiple Search Strategies*”, European Journal of Operational Research, Vol. 247, 3, pp. 732-744.
102. Liou, C. and Hsieh, Y. (2015), “*A Hybrid Algorithm for the Multi-stage Flow Shop Group Scheduling with Sequence-Dependent Setup and Transportation Times*”, International Journal of Production Economics, DOI: 10.1016/j.ijpe.2015.10.002.

103. Liu, Y., Dong, H., Lohse, N. and Petrovic, S. (2016), “*A Multi-objective Genetic Algorithm for Optimisation of Energy Consumption and Shop Floor Production Performance*”, International Journal of Production Economics, DOI: 10.1016/j.ijpe.2016.06.019.
104. Liu, Y., Yin, M. and Gu, W. (2014), “*An Effective Differential Evolution Algorithm for Permutation Flow Shop Scheduling Problem*”, Applied Mathematics and Computation, Vol. 248, pp. 143-159.
106. Loukil, T., Teghem, J. and Fortemps, P. (2007), “*A Multi-Objective Production Scheduling Case Study Solved By Simulated Annealing*”, European Journal of Operational Research, Vol. 179, 3, pp. 709-722.
107. Lu, H. and Huang, Y. (2015), “*An Efficient Genetic Algorithm with a Corner Space Algorithm for a Cutting Stock Problem in the TFT-LCD Industry*”, European Journal of Operational Research, Vol. 246, 1, pp. 51-65.
108. Martin, S., Ouelhadj, D., Beullens, P., Ozcan, E., Juan, A.A. and Burke, E.K. (2016), “*A Multi-Agent Based Cooperative Approach TO Scheduling and Routing*”, European Journal of Operational Research, DOI: 10.1016/j.ejor.2016.02.045
109. Massim, Y., Yalaoui, F., Amodeo, L., Chatelet, E. and Zeblah, A. (2010), “*Efficient Combined Immune-Decomposition Algorithm for Optimal Buffer Allocation in Production Lines for Throughput and Profit Maximization*”, Computers & Operations Research, Vol. 37, 4, pp. 611-620.
110. Matai, R. (2015), “*Solving Multi Objective Facility Layout Problem by Modified Simulated Annealing*”, Applied Mathematics and Computation, Vol. 261, pp. 302-311.
111. Metropolis, N. ; Rosenbluth, A. W. , Rosenbluth, M. N., Teller, A.H. and Teller, E. (1953), "Equation of State Calculations by Fast Computing Machines", The Journal of Chemical Physics, Vol. 21, 6, pp. 1087.
112. Mika, M., Waligora, G. and Weglarz, J. (2008), “*Tabu Search for Multi-Mode Resource-Constrained Project Scheduling with Schedule-Dependent Setup Times*”, European Journal of Operational Research, Vol. 187, 3, pp. 1238-1250.
113. Minghe, S. (2002), “*The Transportation Problem with Exclusionary Side Constraints and Two Branch-and-Bound Algorithms*”, European Journal of Operational Research, Vol. 140, 3, pp. 629-647.

114. Mohammad, M. (2014), “*A Novel Hybrid Genetic Algorithm to Solve the Sequence-Dependent Permutation Flow-Shop Scheduling Problem*”, International Journal Advanced Manufacturing Technology, Vol. 71, pp. 429–437.
115. Molitor, P (1985),”*Layer Assignment by Simulated Annealing*”, Microprocessing and Microprogramming, Vol. 16, 4, pp. 345-349.
116. Morini, M. and Pellegrino, S. (2016), “*Personal Income Tax Reforms: A Genetic Algorithm Approach*”, European Journal of Operational Research.
117. Moslehi, G., Mirzaee, M., Vasei, M., Modarres, M. and Azaron, A. (2009), “*Two-Machine Flow Shop Sheduling to Minimize the Sum of Maximum Earliness and Tardiness*”, International Journal of Production Economics, Vol. 122, 2, pp. 763-773.
118. Mousavi, S. M., Mousakhani, M. and Zandieh, M. (2013), “*Bi-Objective Hybrid Flow Shop Scheduling: A New Local Search*”, International Journal of Advanced Manufacturing Technology, Vol 64, pp. 933–950.
119. Mousavi, S. M., Mousakhani, M. and Zandieh, M. (2013), “*Bi-objective Hybrid Flow Shop Scheduling: A New Local Search*”, International Journal Advanced Manufacturing Technology, Vol. 64, pp. 933–950.
120. Mu, X., Zhang, Y. and Liu, S. (2007), “*A New Branch and Bound Method with Pretreatment for the Binary Quadratic Programming*”, Applied Mathematics and Computation, Vol. 192, 1, pp. 252-259.
121. Muppani, V. R. and Adil, G. K. (2008), “*Efficient Formation of Storage Classes for Warehouse Storage Location Assignment: A Simulated Annealing Approach*”, Omega, Vol. 36, 4, pp. 609-618.
122. Navaei, J., Ghomi , S.M.T.F., Jolai, F. and Mozdgir, A. (2014), “*Heuristics for an Assembly Flow-Shop with Non-identical Assembly Machines and Sequence Dependent Setup Times to Minimize sum of Holding and Delay Costs.*” Computers & Operations Research, Vol. 44, pp. 52–65.
123. Nearchou, A. C. (2004), “*A Novel Meta-heuristic Approach for the Flow Shop Scheduling Problem*”, Engineering Applications of Artificial Intelligence, Vol. 17, 3, pp. 289-300.
124. Nearchou, A. C. (2005), “*A Differential Evolution Algorithm for Simple Assembly Line Balancing* ”, IFAC Proceedings Volumes, Vol. 38, 1, pp. 247-252.

125. Nowicki E., Zdrzałka, S. (1988), “*A Two-Machine Flow Shop Scheduling Problem with Controllable Job Processing Times*”, European Journal of Operational Research, Vol. 34, 2, pp. 208-220.
126. Ouenniche, J. and Boctor, F. F (2001), “*The Two-Group Heuristic to Solve the Multi-Product, Economic Lot Sizing and Scheduling Problem in Flow Shops*”, European Journal of Operational Research, Vol. 129, 3, pp. 539-554.
127. Ouyang, L. Y. and Chang, C. T. (2013), “*Optimal Production Lot with Imperfect Production Process under Permissible Delay in Payments and Complete Backlogging*”, International Journal of Production Economics, Vol. 144, 2, pp. 610-617.
128. Ozturk, O., Begen, M. A. and Zaric, G. S. (2014), “*A Branch and Bound based Heuristic for Make-span Minimization of Washing Operations in Hospital Sterilization Services*”, European Journal of Operational Research, Vol. 239, 1, pp. 214-226.
129. Pakzad-Moghaddam, S. H. (2015), “*A Levy Flight Embedded Particle Swarm Optimization for Multi-Objective Parallel-Machine Scheduling with Learning and Adapting Considerations*”, Computers & Industrial Engineering, Vol. 9, pp. 109–128.
130. Pan, Q., Tasgetiren, M. F., Suganthan, P. N., Chua, T. J. (2011), “*A Discrete Artificial Bee Colony Algorithm for the Lot-Streaming Flow Shop Scheduling Problem*”, Information Sciences, Vol. 181, 12, pp. 2455-2468.
131. Patel, V. and Savsani, V. (2016), “*Multi-objective Optimization of a Sterling Heat Engine Using TS-TLBO (Tutorial Training and Self learning Inspired Teaching-Learning Based Optimization) Algorithm*”, Energy, Vol. 95, pp. 528-541.
132. Pettit, E. J. and Pettit, M. J. (1987), “*Analysis of the Performance of a Genetic Algorithm-Based System for Message Classification in Noisy Environments*”, International Journal of Man-Machine Studies, Vol. 27, 2, pp. 205-220.
133. Potts C. N. (1980), “*An Adaptive Branching Rule for the Permutation Flow-Shop Problem*”, European Journal of Operational Research, Vol. 5, 1, pp. 19-25
134. Potts, C. N. and Baker, K. R. (1990), “*Flow Shop Scheduling with Lot Streaming*”, Operations Research Letters, Volume 8, 6, pp. 297-303.
135. Prescilla, K. and Selvakumar, A. I. (2013), “*Modified Binary Particle Swarm optimization Algorithm Application to Real-Time Task Assignment in Heterogeneous Multiprocessor, Microprocessors and Microsystems*”, Volume 37, 6–7, pp. 583-589.

136. Qian, B., Wang, L., Huang, D., Wang, W. and Wang, X. (2009) “*An Effective Hybrid De-Based Algorithm for Multi-Objective Flow Shop Scheduling With Limited Buffers*”, *Computers & Operations Research*, Vol. 36, 1, pp. 209-233.
137. Quesada, I. and Grossmann, I. E. (1992), “*An LP/NLP Based Branch and Bound Algorithm for Convex MINLP Optimization Problems*”, *Computers & Chemical Engineering*, Vol. 16, Issue 10, pp. 937-947.
138. Raaymakers, W. H. M. and Hoogeveen, J. A.(2000), “*Scheduling Multipurpose Batch Process Industries With No-Wait Restrictions by Simulated Annealing*”, *European Journal of Operational Research*, Vol. 126, 1, pp. 131-151.
139. Rao, R. V., Savsani, V.J. and Vakharia, D.P. (2011), “*Teaching–Learning-Based Optimization: A novel Method for Constrained Mechanical Design Optimization Problems*” *Computer-Aided Design*, Vol. 43, pp. 303–315
140. Rios-Mercado, R. Z. and Bard, J. F. (1998), “*Computational Experience with A Branch-And-Cut Algorithm for Flow-shop Scheduling with Setups*”, *Computers & Operations Research*, Volume 25, 5, pp. 351-366.
141. Rossi, F. L., Nagano, M. S. and Neto, R. F. T. (2016), “*Evaluation of High Performance Constructive Heuristics for the Flow Shop with Make-span Minimization*”, *International Journal of Advanced Manufacturing Technology*.
142. Roy, P.K., S.P. and Thakur, S.S. “*Optimal VAR Control for Improvements in Voltage Profiles and for Real Power Loss Minimization Using Biogeography Based Optimization*”, *International Journal of Electrical Power & Energy Systems*, Vol. 43, 1, pp. 830–838.
143. Ruiz, R. and Stutzle, T. (2008), “*An Iterated Greedy Heuristic for the Sequence Dependent Setup Times Flow-shop Problem with Make-span and Weighted Tardiness Objectives*”, *European Journal of Operational Research*, Vol. 187, 3, pp. 1143-1159.
144. Ruiz, R., Maroto, C. and Alcaraz, J. (2005), “*Solving the Flow-shop Scheduling Problem with Sequence Dependent Setup Times Using Advanced Metaheuristics*”, *European Journal of Operational Research*, Vol. 165, 1, pp. 34-54.
145. Santosa, B. and Safitr, A. L. (2015),” *Biogeography-based Optimization (BBO) Algorithm for Single Machine Total Weighted Tardiness Problem (SMTWTP)*”, *Procedia Manufacturing* ,Vol. 4, pp. 552-557.

146. Saraswat, A., Venkatadri, U., Castillo, I. (2016), “*A Framework for Multi-Objective Facility Layout Design*”, Computers & Industrial Engineering. DOI: 10.1016/j.cie.2015.09.006
147. Scudder, G. D. and Hoffmann, T. R. (1985), “*An Evaluation of Value Based Dispatching Rules in a Flow Shop*”, European Journal of Operational Research, Vol. 22, 3, pp. 310-318.
148. Sendova-Franks, A. and Franks, N. R. (1993), “*Task Allocation in Ant Colonies within Variable Environments (A Study of Temporal Polytheism): Experimental*”, Bulletin of Mathematical Biology, Vol. 55, 1, pp. 75-96.
149. Shaabani, H. and Kamalabadi, I. N. (2016), “*An Efficient Population-Based Simulated Annealing Algorithm for the Multi-Product Multi-Retailer Perishable Inventory Routing Problem*”, Computers & Industrial Engineering, Vol. 99, pp. 189-201.
150. Shahsavar, A., Najafi, A.A., Niaki, S.T.A. (2015), “*Three Self-Adaptive Multi-Objective Evolutionary Algorithms for a Triple-Objective Project Scheduling Problem*”, Computers & Industrial Engineering, Vol. 87, pp. 4–15.
151. Shyu, S. J., Lin, B. M. T and Yin, P. Y. (2004) “*Application of Ant Colony Optimization for No-Wait Flow-shop Scheduling Problem to Minimize the Total Completion Time*”, Computers & Industrial Engineering, Vol. 47, 2–3, pp. 181-193.
152. Simon, D. (2008), “*Biogeography-Based Optimization*”, IEEE Transactions on Evolutionary Computation, Vol. 12, 6, pp. 702-713.
153. Singh, M. R. and Mahapatra, S. S. (2012), “*A Swarm Optimization Approach for Flexible Flow Shop Scheduling with Multiprocessor Tasks*” ,International journal of Advances Manufacturing and Technology, Vol. 62, pp. 267–277.
154. Singh, M.R. and Mahapatra, S.S. (2012), “*A Swarm Optimization Approach For Flexible Flow Shop Scheduling With Multiprocessor Tasks.*” International Journal of Advanced Manufacturing Technology, Vol. 62, pp. 267–277.
155. Smutnicki, C., Pempera, J., Rudy, J. and Zelazny, D., (2015), “*A New Approach for Multi Criteria Scheduling*”, Computers & Industrial Engineering, Vol. 90, pp. 212–220.
156. Song, Y. H. Chou, C. S. and Stonham, T. J. (1999), “*Combined Heat and Power Economic Dispatch by Improved Ant Colony Search Algorithm*”, Electric Power Systems Research, Vol. 52, 2, pp. 115-121.

157. Souza, S. S. F., Romero, R., Pereira, J. and Saraiva, J. T. (2016), “*Artificial Immune Algorithm Applied to Distribution System Reconfiguration with Variable Demand*”, *International Journal of Electrical Power & Energy Systems*, Vol. 82, pp. 561-568.
158. Stark, A. E. (1990), “*An Algorithm for Computing Standard Errors from Categorical Data*”, *Computational Statistics & Data Analysis*, Vol. 10, 3, pp. 293-296.
159. Sukkerd, W. and Wuttiornpun, T. (2016), “*Hybrid Genetic Algorithm and Tabu search for finite Capacity Material Requirement Planning System in Flexible Flow Shop with Assembly Operations*”, *Computers & Industrial Engineering*, Vol. 97, pp. 157-169.
160. Tang, D., Dai, M., Salido, M.A., and Giret, A. (2016), “*Energy-Efficient Dynamic Scheduling For A Flexible Flow Shop Using An Improved Particle Swarm Optimization*”, *Computers in Industry*, Vol. 81, pp. 82-95.
161. Tavakkoli-Moghaddam, R., Rahimi-Vahed, A. and Mirzaei, A. H. (2007), “*A Hybrid Multi-Objective Immune Algorithm for a Flow Shop Scheduling Problem with Bi-Objectives: Weighted Mean Completion Time and Weighted Mean Tardiness*”, *Information Sciences*, Vol. 177, 22, pp. 5072-5090.
162. T'kindt, V., Monmarche, N., Tercinet, F. and Laugt, D. (2002), “*An Ant Colony Optimization Algorithm to Solve a 2-machine Bi-Criteria Flow-shop Scheduling Problem*”, *European Journal of Operational Research*, Vol. 142, 2, pp. 250-257.
163. Toledo, C.F.M., Resende, R., Oliveira, R. and Franca, P.M. (2013), “*A Hybrid Multi-Population Genetic Algorithm Applied to Solve the Multi-level Capacitated Lot Sizing Problem with Backlogging*” *Computers & Operations Research*, Vol. 40, pp. 910–919.
164. Tyagi, N., Varshney, R.G. and Chandramouli, A.B. (2013), “*Six Decades of Flow Shop Scheduling Research*”, *International Journal of Scientific & Engineering Research*, Volume 4, 9, pp. 854-864.
165. Uetake, O., Tsubone, H. and Ohba, M. (1995), “*A Production Scheduling System in a Hybrid Flow Shop*”, *International Journal of Production Economics*, Vol. 41, 1, pp. 395-398.
166. Unler, A. and Murat, A. (2010), “*A Discrete Particle Swarm Optimization Method for Feature Selection in Binary Classification Problems*”, *European Journal of Operational Research*, Vol. 206, 3, pp. 528-539.
167. Vanchipura, R. and Sridharan, R. (2013), “*Development and Analysis of Constructive Heuristic Algorithms for Flow Shop Scheduling Problems with Sequence-Dependent Setup*”

- Times.*” International Journal of Advanced Manufacturing Technology, Vol. 67, pp. 1337–1353.
168. Varadharajan, T. K. and Rajendran, C. (2005), “*A Multi-Objective Simulated-Annealing Algorithm for Scheduling in Flow-shops to Minimize the Make-span and Total Flow Time of Jobs.*”, European Journal of Operational Research, Vol. 167, 3, pp. 772-795.
169. Velez-Gallego, M. C., Maya, J. and Montoya- Torres, J. R. (2016), “*A Beam Search Heuristic for Scheduling a Single Machine with Release Dates and Sequence Dependent Setup Times to Minimize the Make-span*”, *Computers and Operation Research*, DOI: 10.1016/j.cor.2016.04.009.
170. Wang, D., Fung, R. Y. K. and Ip, W. H. (2009), “*An Immune-Genetic Algorithm for Introduction Planning of New Products*”, *Computers & Industrial Engineering*, Vol. 56, 3, pp. 902-917.
171. Wang, J. and Zhang, B. (2015), “*Permutation Flow-shop Problems with Bi-criterion Make-span and Total Completion Time Objective and Position-Weighted Learning Effects* , *Computers & Operations Research*, Vol. 58, pp. 24–31.
172. Wang, X. and Duan, H. (2014), “*A Hybrid Biogeography-Based Optimization Algorithm for job Shop Scheduling Problem*”, *Computers & Industrial Engineering*, Vol. 73, pp. 96–114.
173. Wen, X. and Song, A. (2004), “*An Immune Evolutionary Algorithm for Sphericity Error Evaluation*”, *International Journal of Machine Tools and Manufacture*, Vol. 44, 10, pp. 1077-1084.
174. Wille, E.C.G., Yabcznski, E. and Lopes, H. S. (2011) “*Discrete Capacity Assignment in IP Networks using Particle Swarm Optimization*”, *Applied Mathematics and Computation*, Vol. 217, 12, pp. 5338-5346.
175. Wu, T., Shi, L., Geunes, J. and Akartunal, K. (2011), “*An Optimization Framework For Solving Capacitated Multi-Level Lot-Sizing Problems With Backlogging*”, *European Journal of Operational Research*, Vol. 214, 2, pp. 428-441.
176. Xiong, F., Xing, K., Wang, F., Lei, H. and Han, L. (2014), “*Minimizing the Total Completion Time in a Distributed Two Stage Assembly System with Setup Times*”, *Computers & Operations Research*, Vol. 47, pp. 92–105.

177. Xu, Z., Sun, L. and Gong, J. (2008), “*Worst-case Analysis for Flow Shop Scheduling with a Learning Effect*”, International Journal of Production Economics, Vol. 113, 2, pp. 748-753.
178. Yagmahan, B. and Yenisey, M.M. (2010), “A Multi-Objective Ant Colony System Algorithm For Flow Shop Scheduling Problem”, Expert Systems with Applications, Vol. 37, 2, pp. 1361-1368.
179. Yalaoui, N., Amodeo, L., Mahdi, H. and Yalaoui, F. (2009), “*Hybrid Method to Solve a Facility Layout Problem: Genetic Algorithm-Particle Swarm Optimization*”, IFAC Proceedings Volumes, Vol. 42, 4, pp. 1251-1255.
180. Yamada, T., Takahashi, H. and Kataoka, S. (1997), “*A Branch-and-Bound Algorithm for the Mini-Max Spanning Forest Problem*”, European Journal of Operational Research, Vol. 101, 1, pp. 93-103.
181. Yan, T., Rabinovich, E., Dooley, K. and Evers, P.T. (2010), “*Managing Backlog Variation in Order Fulfillment: The Case of Internet Retailers*”, International Journal of Production Economics, Vol. 128, 1, pp. 261-268.
182. Yin, P. Y and Wang, J. Y. (2006), “*A Particle Swarm Optimization Approach to the Nonlinear Resource Allocation Problem*”, Applied Mathematics and Computation, Vol. 183, 1, pp. 232-242.
183. Yu, Y., Tang, J., Gong, J., Yin, Y. and Kaku, I. (2014), “*Mathematical Analysis and Solutions for Multi-Objective Line-Cell Conversion Problem*”, European Journal of Operational Research, Vol. 236, pp. 774–786.
184. Zandieh, M., Ghomi, S. M. T. F. and Hussein, S. M. M. (2006), “An Immune Algorithm Approach to Hybrid Flow Shops Scheduling with Sequence-Dependent Setup Times”, Applied Mathematics and Computation, Volume 180, 1, pp. 111-127.
185. Zhang, C., Sun, J., Zhu, X. and Yang, Q. (2008) “*An Improved Particle Swarm Optimization Algorithm for Flow-shop Scheduling Problem*”, Information Processing Letters, Vol. 108, 4, pp. 204-209.
186. Zhao, F., Li, S., Sun, J. and Mei, D. (2009) “*Genetic Algorithm for the One-Commodity Pickup-and-Delivery Traveling Salesman Problem*”, Computers & Industrial Engineering, Vol. 56, 4, pp. 1642-1648.

187. Zhou, S., Liu, M., Chen, H. and Li, X. (2016), “*An Effective Discrete Differential Evolution Algorithm for Scheduling Uniform Parallel Batch Processing Machines with Non-Identical capacities and arbitrary job sizes*”, *International Journal of Production Economics*, Vol. 179, pp. 1-11.

188. Zou, F., Wanga, L, Hei, X, Chen, D. and Wanga, B. (2013), “*Multi-objective Optimization Using Teaching-Learning-Based Optimization Algorithm*”, *Engineering Applications of Artificial Intelligence*, Vol. 26, pp. 1291–1300.