DESIGN OF A NOVEL HYBRID HEURISTICS DISTRIBUTION 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

Kulbir Singh
(Reg. No. 11617178)

Under the Guidance of

Harpreet Singh
(Assistant Professor)
UID: 12531

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA (DISTT. KAPURTHALA), PUNJAB

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

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 Distribution 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 embodied in this dissertation has not been submitted in part or full to any other University or Institute for the award of any degree.

Date: **(Kulbir Singh)**

Reg. No. 11617178

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 internal viva-voce examination of Kulbir Singh, has been held on ______________________ .

Signature of Examiner

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PB, INDIA

CANDIDATE'S DECLARATION

I, Kulbir Singh, Reg. No. 11617178 hereby declare that the work presented entitled **"DESIGN OF A NOVEL HYBRID HEURISTICS DISTRIBUTION 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 2017 to NOV 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:

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 Dr. M.S. Saini, 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.

Kulbir Singh

Reg. No. 11617178

LIST OF FIGURES

NOMENCLATURE

ACO	Ant Colony Optimization
AIS	Artificial Immune System
ANOVA	Analysis of Variance
BBO	Biogeography- Based Optimization
B&B	Branch and bound
BRILS	Biased-Randomized Iterated Local Search
BS-HH	Backtracking Search Hyper-heuristic
BOLS	Bi-Objective Local Search Algorithm
BSA	Backtracking Search Algorithm
CLSP	Capacitated Lot Sizing and Scheduling Problem
COP	Combinatorial Optimization Problem
CMA	Competitive memetic algorithm
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
FFS	Fuzzy Flowshop scheduling
FSS	Flow Shop Scheduling
GA	Genetic Algorithm
GWO	Grey Wolf Optimizer
GVNS	General Variable Neighborhood Search
HBBO	Hybrid Biogeography Optimization Algorithm
HBSA	Hybrid Backtracking Search Algorithm
HFSPB	Hybrid Flow Shop Parallel Batching
HFSMT	Hybrid Flow Shop Scheduling with Multiprocessor Task
HGA	Hybrid Genetic Algorithm
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
LOV	Largest Order Value
LSL	Least Slack Time
MACS	Multi-Ant Colony System
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
MOACSA	Multi-Objective Ant-Colony System Algorithm
MPD	Maximum Percentage Deviation
MMOIG	Modified Multi-Objective Iterated Greedy Algorithm
MODE	Multi-objective Differential Evolutionary
MOPSO	Multi- Objective Particle Swarm Optimization
MOSA	Multi-Objective Simulated Annealing
MONEH	Multi-Objective NEH Algorithm
MOSLS	Multi-Objective Stochastic Local Search
NEH_RMB	Nawaz Enscore Ham and Rio Mercado and Brad Heuristic
NRGA	Non-Dominated Ranked Genetic Algorithm
NSGA	Non-Dominated Sorting Genetic Algorithm
PCB	Printed-Circuit-Board
PEM	Prediction Error Method
PFSP	Permutation Flow Shop Scheduling Problem
PBS	Percentage of the Best Solutions
PH	Polynomial Time Heuristic

PSO	Particle Swarm Optimization
PTS	Parallel Tabu Search
SA	Simulated Annealing
SAA	Simulated Annealing Algorithm
SAL	Sided Assembly Line
SDST	Sequence Dependent Setup Times
SGA	Smart Decoding-Based Genetic Algorithm
SPEA	Strength Pareto Evolutionary Algorithm
SLPSO	Self-Learning Particle Swarm Optimization
SPT	Shortest Processing Time
SS	Scatter Search
TLBO	Teacher Learning Based Optimization
TS	Tabu Search
TSAFSP	Two-Stage Assembly Flow-Shop Scheduling Problem
TP-GA	Two-Phase Genetic Algorithm
TTT	Total Throughput Time
VND	Variable Neighborhood Descent
VNS	Variable Neighborhood Search
PT	Processing Time
TS	Setup Time
[C]	Minimize Cycle Time
[Z]	Minimize Process Time
C_j	Completion Time, where j represents job
P_n	Population size
T_F	Teaching factor
T_j	Tardiness, where j represents job

CONTENTS

CHAPTER 1: INTRODUCTION

1.1 SCHEDULING

In the authentic-time scenario, there subsist many situations in manufacturing system like due date changes, unexpected job release, machine breakdowns and more preponderant processing times, than estimated and expected. The production cost aggregates to high proportion of any firm's expenditure, hence every firm endeavor to get a felicitous design of shop and scheduling of jobs on sundry machines to optimize the task times for long-standing and temporary goals. Scheduling, hence, is a non-ignorable aspect of every manufacturing system. Scheduling is the allocation of inhibited resources (man and machinery), by organizing, controlling and optimizing a sundry set of events in a manufacturing process in a concrete duration. Basically, it validates the production capability when to make, on which equipment and with which staff. Johnson (1954) considered two and three-stage production system which involved set-up times. Panwalkar and Woollam (1980) studied the ordered flow shop sequencing problem with no in-processing waiting (OFSNW) with an aim to diminish the mean flow time. Karabati et al. (1992) investigated the (PFS) problem and addressed as (CSP) problems with finite buffer capacities. A (B&B) approach of the problem was developed, which is easily capable to solve cyclic problem for the production flow line. Deng and Wang (2016) proposed (CMA) algorithm to resolve multi-objective distributed (PFS) problem known as (MODPFSP) under objectives to reduce makespan and total tardiness. Samarghandi and Behroozi (2017) addressed the (FSS) problem where the processing is continuous with due date constraints and makespan criterion. Scheduling is an efficacious method to orchestrate the sequence of tasks and is applicable to accommodation industry, electronic industry, project control, computer science, foodstuffs 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 act processing time and set-up time.

(ii) Ready time: It is the variance between the entrance 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. If the completion time exceeds the due date than there is a delay.

(iv) Completion Time: It is the time at which the job 'j' is actually completed its all sequence and get finished.

(v) Flow time: It is the amount of time that job 'j' spends in a system. It is variance 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 degree 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 circumstances 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 encompassed in its processing time.
3. Job descriptors are to know in advance.
4. One machine is incessantly accessible and never kept idle.
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 numerous tasks associated with sundry jobs, to optimize some performance measures. Figure 1.1 depicts classification of scheduling problem.

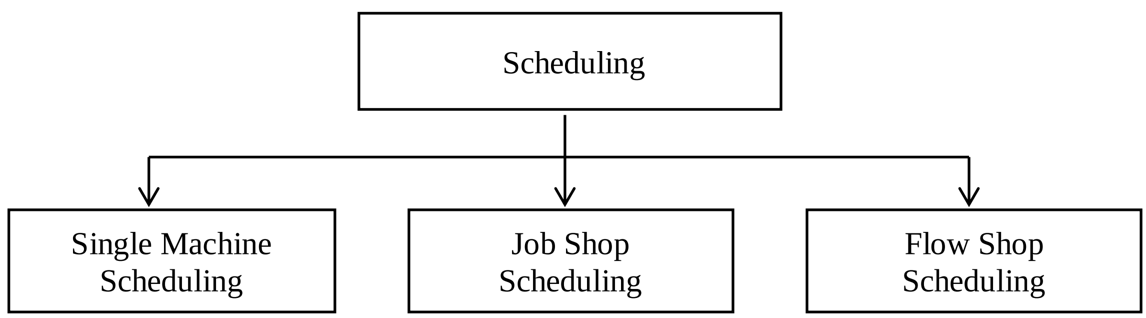

Figure 1.1: Depiction of Classification Scheduling Problem

(i) Single machine scheduling

A Single Machine scheduling, comprises of 'n' independent jobs, each with solo operation.

The objective of diminishing the makespan is achieved by positioning the jobs on the source of in-process time, suggesting that, jobs with a lesser amount of in-process time are placed ahead of those with higher in-process times.

(ii) Job Shop Scheduling

In job shop scheduling, each job has ‘m’ different operations. If jobs are having less than m operations, we assume required number of pretend operations with zero process times, ensuring the condition of identical number of operations. Taillard (1993) discussed the various problems for basic model to lessen the make-span. The processing of jobs respective to their sequences are not same. Hence, the flow of individual job is not unidirectional and it is not compulsory to process all the jobs on each machine.

(iii) Flow shop scheduling (FSS)

In flow shop environment, each job has to go through a series of operations in the identical order, implicatively insinuating that the jobs have to follow the identical route or process sequence, but the time of processing on each operation on a job will be different from that of other jobs. Rock (1983) studied two machine flowshop scheduling for minimizing maximum lateness and mean flow time with no wait in process conditions. A flow-shop scheduling quandary encompasses scheduling ‘n’ jobs on ‘m’ machines. A job consists of ‘m’ tasks and the j^{th} operation on every job must be process on the j^{th} machine, which only transpires if it has already consummated on machine $j-1$ and machine j is idle. Ashour (1970) proposed a B&B algorithm I, in which a new lower bound was developed for getting the best results under makespan criterion. This lower bound is helps in resolving the struggle of jobs on the last machine. Reddi and Ramamoorthy (1972) considered the (FSS) problem with no middle storage between the machines. The problem is consequential in computer systems as firstly it is a footstep towards the flow-(FSS) problem with no finite middle storage and secondly it is profoundly utilizable in sizably voluminous computer systems where huge buffers are unreasonable and due to the elimination of intricate supervisory systems to enforce opportune routing of waiting jobs. Potts (1980) framed a B&B algorithm for permutation flow-shop problem, which included dominance rules, and computed upper bounds to abate the maximum total flow time. Grabowski et al. (1983) demonstrate a B&B algorithm for a two-machine scheduling problem entailing Release and Due Dates to Diminish Maximum Lateness. Lowe bound are used for obtaining optimal results. The Jackson's rule was applied for strengthening these Lower bounds. Szwarc (1988) considered a simplification of the classical ‘n’ jobs ‘m’ machine problem where the n items, are grouped into k immovable sequences (clusters) and are processed on m machines. A same order complexity is being sought that decreases the accomplishment time of processing all items. Panwalkar (1991) addressed two-machine (FSS) problem with travel time between machines. This flowshop problem contains a single transporter (called the AGV) that carries jobs from one machine to other machine and there is limitless buffer space between the machines. A constructive algorithm was proposed that diminishes the makespan objective of the problem. Liao (1993) studied a premutation based (FSS) problem with a goal to lessen makespan and number of machine idle intervals. The advantage of minimizing the number of machine idle interval, reduces the necessary time to restart machines. Riezebos et al. (1995) deliberated the (FSS) problem with numerous operations and time lags. In the FSS problem there is one machine in each stage and multiple operations are performed on the jobs at each stage. Also, there is continues sequence order in the stages that is when one job get finished then the next job can be processes. Three types of lower bounds are developed: job-based bounds, machine-based bounds and due-date-based bounds and further cast-off in the construction of algorithms for the problem considering makespan criteria. Liu et al. (1997) addressed a problem of FSS where the job processing follows same order and function to minimize production tardiness and on a constraint related to raw materials release. A novel “separable” integer programming was formulated by presenting the virtual sequence, and supplement target function and constraints in terms of virtual sequence variables for the problem. After that lagrangian relaxation procedure was used for answering separable integer programming model. Ronconi and Armentano (2001) purposed lower bounds to study the ‘n’ jobs ‘m’ machine problem with blocking in-process for reduction of makespan followed with total tardiness. Blocking is common in consecutive manufacturing, where no transitional buffer storage is offered. Then by the implementation of lower bounds a B&B algorithm was developed for the problem. Pan and Chen (2003) studied a re-entrant premutation flow-shop (RFS) problem with makespan as criterion. The three models are formulated (Model 1, Model 2 and Model 3) for the problem, also five heuristic algorithms (Heuristic 1, 2, 3, 4 and 5) for the classical permutation flow-shop are modified to solve the RPFS problem. Yanai and Fujie (2005) considered a three-machine (FSS) problem where same processing order is being followed, the criterion of problem is to abate the total completion time without indolent times respectively reducing the makespan on second machine. A B&B algorithm was developed with effective branching rules and dominance properties that decrease the search space. Pan and Wang (2008) proposed an effective hybrid DPSO called as (HDPSO) algorithm for the no-idle permutation (FSS) problem with makespan criterion.

The (DPSO) algorithm is grounded with permutation representation and for the enhancement of exploration and exploitation of the heuristics a local search based on the insert neighborhood was integrated to the heuristics. Ahmadizar and Farahani (2012) proposed a (HGA) algorithm for open shop scheduling to reduce 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. Costa et al. (2014) considered HFS with parallel batching (HFSPB), which involved stages with parallel and proposed a (MIP) model with a GA, which uses crossover operator for scheduling the jobs to reduce the make-span. Fung et al. (2016) considered a two-stage FSS problem with a buffer. Some assumption is considered in the problem that all operations must have equivalent processing time and one machine can process only one individual job if some amount of buffer space is assigned to the job. The amount of assigned buffer space must be differing from job to job. Shen and Gupta (2017) addressed a family scheduling problem in flowshop manufacturing systems, where batching decisions and non-permutation schedules are taken into consideration. The batch availability assumption was adopted for the problem. A TS algorithm was projected for the problem with makespan criteria. Kampmeyer et al. (2016) considered a synchronous FSS problem that consists of two dominating machines. The synchronous flowshop is a non- pre-emptive permutation flow, where exchanges of jobs starting with one machine then onto the next take place at same time. Processing of a job on the following machine may just begin after the present jobs on all machines are done. A solution algorithm was proposed which displaying the problem as a special vehicle routing problem. The problem was formulated into mixed integer liner programming model, further a TS algorithm and lower bounds are provided for the problem.

1.2 HEURISTICS

Heuristics refers to as an approach for the problem resolving, or discovering that engages a practical method not mandatory to be optimal for the problem, but sufficient to obtain results. The heuristics helps in solving multifaceted or incomplete problems by some judgments and decisions followed from nature.

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.

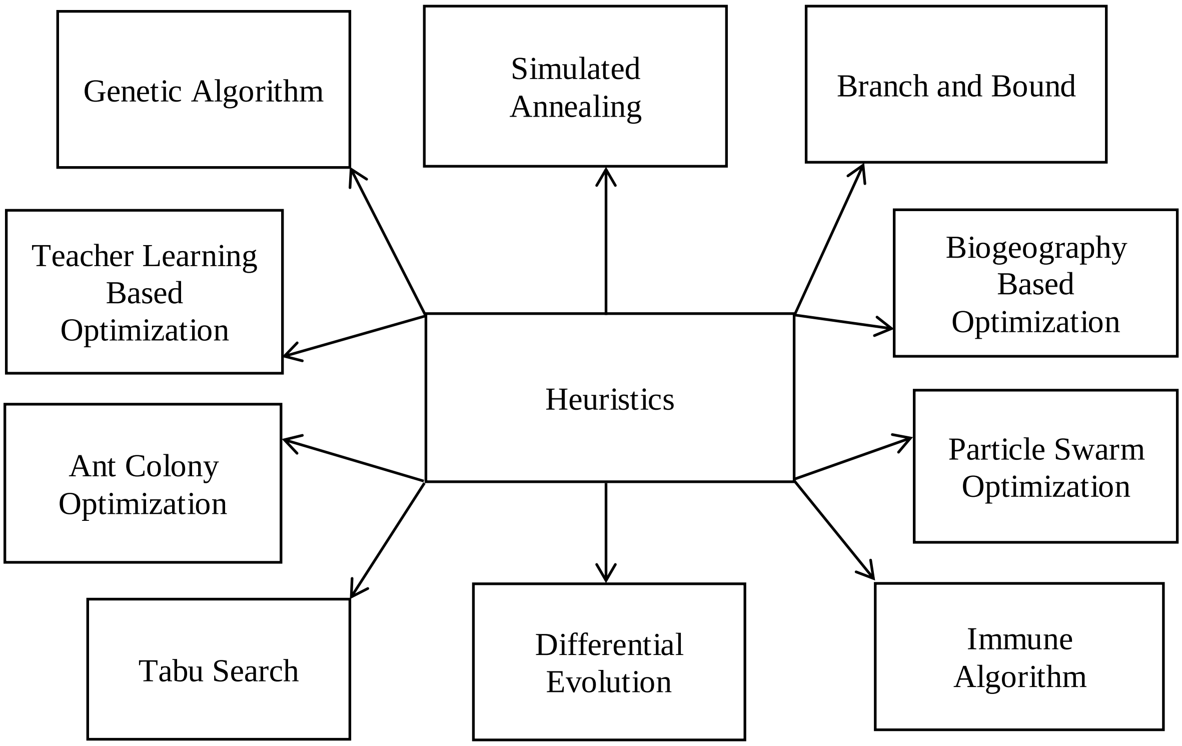

Figure 1.2: Depictions of Various Methods to Solve Scheduling Problems

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 off-springs 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.

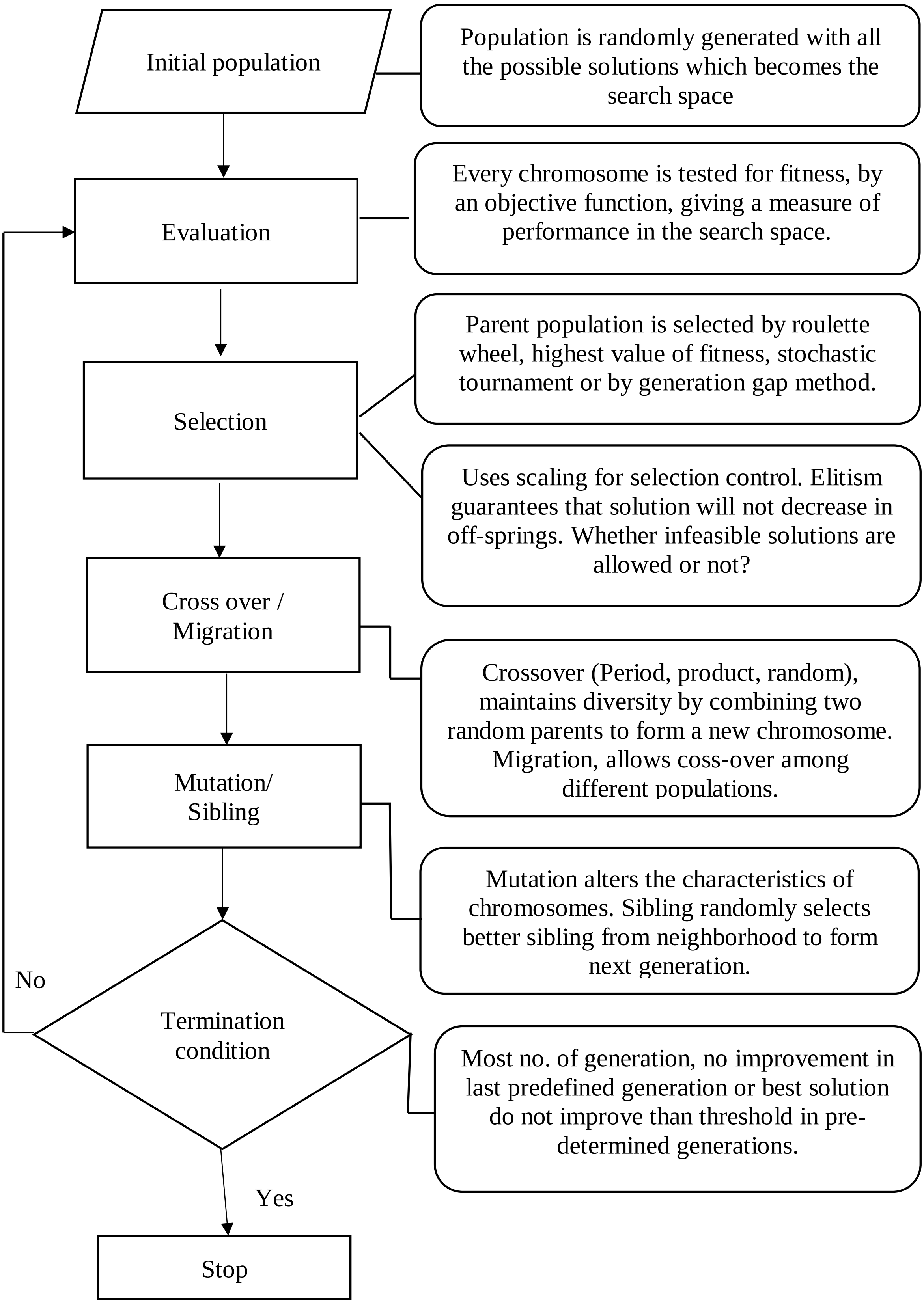

Figure 1.3: Flow Process Chart Depicting Steps in GA for Optimization

Liepins and Hilliard (1989) studied the Genetic algorithm. The GA was constructed through the building block hypothesis along with the implicit parallelism mechanisms and they are important for its success. Also, discuss the difficulties occurring during the implementation of genetic algorithm. There are variety of applications of GA which was studied, and for each application GA provided good solution capabilities which has been not provide by other methods. Mirabi (2014) studied the (FSS) problem in sequence-dependent circumstances for a massive number of real-world problems. A meta-heuristic approach based on genetic algorithm called HGA was developed to solve the problem. The proposed HGA implement a modified NEH_RMB approach to generate the population of initial chromosomes and also use a better-quality heuristic called as the (ISP) to improve them. 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. Low et al. (2004) presented a modified NEH algorithm as a preliminary solution searching algorithm for the (FSS) problems with minimizing makespan objective. Also, proposed a modified (SA) searching procedure which consists of ‘‘restarting solution mechanism’’ and some additional conditions for improving the efficiency of the solutions.

1.2.4 Branch and Bound Method

Branch and Bound (B&B) proposed by Land and Doig in 1960’s, and is an algorithm to discover 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. McMahon and Burton (1967) proposed B&B Method for the three machine (FSS) problem. The job- based bounds are added for increasing the efficiency of the B&B

algorithm. Also, a simple rule of ordering the jobs by ascending values can lead to improvement of 10 percent in the algorithm.

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 Gamberdella (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 follows 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. Yagmahan and Yenisey (2010) formulated a (MOACSA) algorithm for the (FSS) problem with multi-objectives of makespan and total flowtime. The projected algorithm was formed by the combination of ACO approach and a local search strategy.

1.2.6 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a meta-heuristic approach proposed by Kennedy and Eberhart (1995) for optimization of constant non-linear functions. PSO was enthused by the kinetics of a drove of birds probing for food, each bird named a particle regulates its probing direction affording to two factors, its own finest earlier knowledge and the best understanding of all other members. Liao et al. (2012) developed a new method by Hybridizing (PSO) algorithm with a bottleneck heuristic for the(HFS) problem. Geng et al. (2016) proposed SSPSO algorithm for the (E/T) FSS problem with uncertain processing time and distinct due window. The algorithms were formed by integrating the scatter search (SS) algorithm into (PSO) algorithm.

1.2.7 Tabu Search

Tabu Search (TS) is a meta-heuristic method engendered by Glover (1986) and formalized in (1989). The TS is a local search method used for optimization of problems. It approaches the neighborhood solution, which implies the comparable arrangements aside from with minor details to produce another enhanced arrangement. Faculty to accept the worsening moves on sub-optimal and plateaus, where arrangements are probably going to fit similarly and, acquainting the proscriptions to stop the search from returning to yester visited arrangements, enhances its performance. Yip et al. (2005) addressed a problem of (FSS) with setup time. In the problem the processing and removal times are parted with objective to diminution makespan. heuristics based on the tabu search method was presented, during foundation the heuristic construct artificial processing times for operations. An improved NEH heuristics was used for verdict initial solutions, further enhancements in the solutions done by tabu search procedure.

1.2.8 Differential Evolution

Differential Evolution (DE) is a meta-heuristic, and an optimization method proposed by Storn and Price (1997). It is a simple yet puissant population-predicated stochastic search technique for authentic parameter optimization. In DE, incipient candidates are engendered by mutation and crossover operators and a one-to-one competition scheme avariciously deciding whether the incipient candidates will survive in the next generation. Deng and Gu (2012) presented (HDDE) algorithm for (PFS) problem in no-wait environment. A speed up method evaluate the job premutation in the neighborhood technique for reducing the computational complexity. Also, a perturbed local search has imbedded in the HDDE algorithm for increasing its performance. The objective of problem is to reduce the makespan. Zhou et al. (2016) constructed a mathematical model of the reentrant flowshop scheduling with aim to reduce total weighted completion time Then a hybrid differential evolution (DE) algorithm that uses an ensemble model (eEDA), named DE–eEDA, was proposed.

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. Moghaddam et al. (2008) presented immune algorithm for (MFSP) problem under no-wait conditions with respect to minimizes the weighted mean completion time and tardiness. The algorithm searches optimal Pareto frontier for the problem. Alisantoso et al. (2003) developed Immune Algorithm (IA) to schedule the flexible flow shop for manufacturing the PCBs (printed circuit board). The algorithm was incorporated with restarting and accelerating mechanisms for discovering the finest solutions of the problem. The restarting mechanism reduces the dominance solution in the population to get better diversity during the search. For checking the optimality of the solutions, the IA was compared with GA.

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. Shao et al. (2017) created a hybrid meta-heuristic predicated on probabilistic teaching-learning mechanism (mPTLM) to resolve no-wait FSS problem called as (NWFSSP). The meta-heuristic contains of four parts, i.e. (1) screening afore class, in which preliminary method that cumulates a modified (NEH) heuristic and the (OBL) was familiarized. (2) Teaching phase, as the teacher to helps learners to more guaranteeing areas, the Gaussian distribution was employed. (3) Learning phase, an incipient designates of communication with crossover was presented. (4) Studying after class, for upgrading the local search capabilities an enhanced speed-up random insert local search based on (SA) was developed.

1.2.11 Biogeography Based Optimization (BBO)

Biogeography optimization is induced from the nature's geographic distributions and proportioning of the biological organisms and was formulated by Dan Simon in (2008). It is a bio-motivated and population based optimization approach where the virtuousness of the habitat is measured by using (HSI). Suitability index variable (SIV) is used for characterizing the attributes of the natural habitat and expressed as one dimension in a solution. The BBO entails two main operators, migration and mutation. The migration operator distributes information between two existing habitats in order to modify SIV, whereas habitat attributes based on a mutation probability is modified by using mutation operator. Lin and Zhang (2016) developed (HBBO) heuristics approach by amalgamate several heuristics and a modified local search mechanism. The HBBO used to investigated distributed assembly (DAPFSP) problem with an objective of optimizing the makespan. The migration phase was inserted with path relinking mechanism and mutation phase utilizes the Insertion-Job local search method for the modification purpose.

1.3 MULTI-OBJECTIVE FLOW SHOP SCHEDULING PROBLEMS (MFSP)

The scheduling problems consists number of objectives which has to be minimized for obtaining respectable results for the problems. The objectives like make-span, flow-time, tardiness, lateness, earliness, achieving due dates; decreasing job disruptions, energy consumption, scheduling costs etc. The problems in which only one objective to be solved called as single objective problem, the problems with two objectives called as bi-objective problems, the problems considered more than two objectives called as multi-objective problems. MFSP problems are more complex and considered as NP-Hard (non-deterministic polynomial time) whose exact solutions do not exist. Yagmahan and Yenisey (2010) proposed a (MOACSA) algorithm for the (FSS) problem with multi-objectives of makespan and total flowtime. The proposed algorithm was formed by the combination of ACO approach and a LS strategy. Torkashvand et al. (2017) formulated the multi-objective (FSS) problem as a (MILP) model in order to diminish the makespan and total tardiness of jobs. Then, a novel (BBO) algorithm was developed. The algorithm utilizes various mechanisms like initialization and elitism operator, rate-calculation, migration and perturbation. Ding et al. (2016) framework two multi-objective optimizations namely (MONEH) and (MMOIG) algorithm. The energy saving method and accelerating methods were implemented in the extended NEH-Insertion Procedure for enhancing its effectiveness. The first-class non-dominated results produced by proposed algorithms help to make a balance between the makespan & carbon emission. The addressed problem was (PFS) scheduling where the carbon emissions & the makespan has to be minimized. Han et al. (2016) proposed a novel multi-objective optimization algorithm using GA to solve blocking lot-streaming (FSS) problem, in which differences among parents and non-dominated solutions are used to design edge operator and local search strategy was active to explore the search space.

1.4 SEQUENCE DEPENDENT SETUP TIMES (SDST)

In scheduling, set-up time makes problem more intricate and comes to play when production changeover is mandatory between the different jobs, taking different durations to set-up on the machine before starting the process. There are two types of structures;

simple, in which set-up is independent of sequences and decisions for precedent times, and intricate, in which set-up time is dependent on both the factors. There exist 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. Szwarc and Liu (1989) constructed an Additive Model for the m machine n job (FSS) problem with SDST. The objective of the scheduling is to minimize makespan. Relative deviation is used to analyzes performance of the results, whereas worst average deviation measures the quality of the approximate solutions. Nagano et al. (2014) purposed a hybrid metaheuristic based on the Evolutionary cluster search (ECS_NSL) to settle the no-wait (FSS) problem with SDST. The ESC_NSL is the formed from the amalgamation of GA with the cluster search. The ECS_NSL works in ally with evolutionary algorithms, and uses the local search procedure. Due to the application of local search procedure there was significant enhancements in the quality of the solution. Sioud et al. (2017) presented an Enhanced Migrating Birds Optimization (EMBO) Algorithm and a new STH heuristic, for resolving the (PFS) scheduling problem with SDST along with objective of Decreasing the Makespan. An adapted neighborhood search technique was used to enhanced the migrating bird optimization based on the swap and forward insertion moves. The STH heuristic is quicker from all the other existing heuristics for small, medium and large instances.

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. Wu et al. (2011) proposed (MILP) model for capacitated multi-level lot sizing problem, constrained by backlogging with an aim to provide lower bound on optimal solutions. Babaei et al. (2014), developed a genetic algorithm for capacitated scheduling and lot sizing problem. The problem consists SDST, setup carryover and backlogging constraints. A lower bound was established to study the involution and determine the near-optimal solutions in plausible computational time.

CHAPTER 2: LITERATURE REVIEW

2.1 INTRODUCTION

This chapter reviews the contributions made towards multi-objective (FSS) problems, (SDST) 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. As flowshop environment is ecumenical practice in the manufacturing industries, hence paramount to ameliorate the productivity to achieve profit-worthy status in the economy. The consequentiality of reviewing lies in the diversity of flow shop quandary, its parameters which on optimizing/minimizing even one of them would engender paramount 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 engender results and the software used to code the algorithms along with the comparisons made with the respective contemporary methods. The prelude chapter showed the spread of flow shop quandaries 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 makespan in the below contributions.

2.2 RESEARCH MODELS

Ziaeifar et al. (2011) presented a mathematical model for a (HFS) environment with makespan (i.e., C_{max}) and processor assigning cost criteria. In the problem it was assumed that there are several parallel identical processors with limitless middle storage between any two progressive stages are assigned to all of the stages in the hybrid flowshop. A new algorithm was proposed to register the principal target work (i.e., C_{max}). Because of NP-hard of the problem, a GA was additionally proposed to take care of the problem issues. The proposed GA regulate the initial sequence of jobs and task of the processors to the stages. The parameters of GA were tuned by using a full-factorial design. The implementation of tuning was done with design-of-experiments (DOE) approach. Results were investigated by the (ANOVA) technique. So as to assess the proficiency and prevalence of the of the proposed GA, 20 test problems were solved and the associated outcomes were compared with the lower bound. The presented model was coded and solved in LINGO. Fattahi et al. (2012) proposed four Heuristic algorithms in view of the essential thought of Johnson algorithm for two-stage assembly (HFS) scheduling problem. The impartial of scheduling was to decrease the completion time. The problem consists of HFS stage and an assembly stage where numerous set of parts designed for the products are manufactured in the HFS stage, and the complete products were assembled through various parts in the assembly stage. the final solution was obtained from the new lower bounds that were introduced in the heuristic algorithm Furthermore, six procedures were offered to schedule parts in HFS stage. The algorithms were verified in different situations for discovering optimal solutions. The presented algorithms were coded in MATLAB 7/10/0/499 (R2010a). The testing was executed on Intel PC Core 2 processor with 1 GB of RAM. Babaei et al. (2014) addressed the capacitated lot sizing and scheduling problem (CLSP) with SDST, setup carryover, and backlogging. A mixed-integer program (MIP) model was formulated. A Genetic algorithm was established by using the lower bounds for the problem with objective to find an optimum lot sizing and scheduling that reduces arrangement, inventory, and backlogging costs, a lower bound was established and equated against GA to measure the precision of the genetic algorithm. The design of experiment (DOE) was used for selecting the best combination. The DOE was run on Expert Design 8.0.6 Software. The ANOVA strategy was utilized for constructive genetic algorithm experiment, the outcomes demonstrated the adequacy of the approach. The GA was framed in MATLAB. The MIP model and lower bound was implemented in Lingo 8.0 on a PC with Core 2 Duo 2.94 GHz. Mousavi et al. (2013) developed a novel bi-objective local search algorithm (BOLS) for hybrid flow shop scheduling problem under just in time environment. The purpose of scheduling was to minimize bi-criteria objectives, total tardiness and make-span. The BOLS algorithm developed with the combination of Local search method and Heuristics, where the local search included three stages. First stage assigned set of jobs are moved to other machines, second stage changes the order of machines and third stage at the same time changes order of jobs and job set of machines. The triangle method and hull approach verified the quality of solutions, and results were juxtaposed with multi-objective simulated annealing (MOSA) and bi-objective heuristic (BOH) approach. The presented method was applicable to find optimum solutions and to find Pareto frontier using other evolutionary algorithms. The algorithms were framed in MATLAB 7 and implemented on CPU 800 processor having 512 RAM. Mirabi (2014) developed a novel (HGA) algorithm for the sequence-dependent (PFS) scheduling problem with an aim to minimize the makespan. The proposed algorithm uses three genetic operators named as, order crossover, heuristic mutation and inversion mutation. Iterated Swap Procedure (ISP) was to generate an improved population of chromosomes in the algorithm, to produce initial population of chromosomes the GA was amalgamate with modified NEH_RMB approach. The presented HGA was compared with some new heuristics like prediction error method (PEM), polynomial time heuristic (PH) and stochastic hybrid heuristic. The avg, min, and max performance measure (PM) values for each algorithm are calculated and ANOVA approach was utilized to get better results. MATLAB software was used for coding the algorithm and experiment was conducted on Pentium III 1.2 Hz CPU with 512 MB RAM. Komaki and Kayvanfar (2015) considered two-stage (AFS) problem where the first consists ‘m’ parallel identical machines called as “fabrication stage” while the second stage is assembly stage. A novel meta-heuristic (GWO) algorithm was developed, along with numerous heuristic procedures, dispatching rules, along with a lower bound are also developed. The objective of the scheduling was to reduce makespan. Also, a local search was included inside the algorithm to enhance its performance. The execution of the lower bound are assessed by deviation of the LB (DVL) from the best solution of the algorithms, and the performance of algorithm was calculated by relative percentage deviation (RPD). Statistical analysis (ANOVA) was carried out to measure the effectiveness of the proposed LB and working algorithms. MATLAB 7.5.0 was used to encode all the algorithms and executed on Intel Core 2 Duo processor 2.66 GHz PC with 3GB RAM memory. Xiao et al. (2015) studied the non-permutation problem of (FSS) with order acceptance and weighted tardiness (FSS-OAWT). A (LMIP) Model was developed for small-Sized problems and resolved in AMPL/CPLEX a optimization solvers. Also, a (NIP) Model for large-Sized problems was developed and solved by heuristic algorithms. Some theorems for permutation and non-permutation conditions are provided and the properties of FSS-OAWT are investigated. A two-phase GA (TP-GA) was developed for solving the NIP model. These properties and theorems are used in the genetic operators of TP-GA for more effectual search. The TP-GA algorithm was framed in Visual C++ 6.0 and tested on 3.4GHz Intel PC. Torkashvand et al. (2017) formulated the (MFSP) scheduling problem as a MIP model for the makespan and total tardiness ceritions. Then, a novel biogeography-based optimization (BBO) was developed. The algorithm utilizes various mechanisms like initialization and elitism operator, rate-calculation, migration and perturbation. For measuring the performance, the BBO was compared with MOGA, NSGAII and MOSA algorithms. Three performance indicators, dominance ranking, hypervolume and binary ϵ- indicators are used to analyze the experimental results. The means of statistical tests of ANVOA and LSD where carried out to analyze the results performances. The experiments were conducted on a 2.0 GHz Intel Core 2 Duo PC with 2GB of RAM memory, all the algorithms were executed in MATLAB software. Ferrer et al. (2016) studied a Combinatorial Optimization Problem (COP) with non-smooth objective functions, where a new variant of Permutation Flow-Shop Problem (PFSP) was considered and then formulated into a mathematical model. The non-smooth objective functions for the problem are makespan cost and failure risk costs due to continuous process of machines. After that, a (BRILS) algorithm was proposed as a solving approach for the problem. NEH heuristic was used to produce initial solutions for the problem. By implementing these solutions, the algorithm able to produce a number of alternative decent solutions without necessitating a multifaceted setting of parameters. The outcomes demonstrate that by implementing this approach savings can be gained (average *gap* = −1.14%) even for reasonable stages of failure-risk costs. The algorithm was implemented in Java language on a desktop PC with operating system ubuntu 14.04, intel CPU: i7 3.4 GHz processor with 8GB of RAM. Ding et al. (2016) framework two multi-objective optimizations namely multi-objective NEH algorithm (MONEH) and a modified multi-objective iterated greedy (MMOIG) algorithm. The energy saving method and accelerating methods were implemented in the extended NEH-Insertion Procedure for enhancing its effectiveness. The superior non-dominated results produced by the algorithms help to make a balance between the makespan & carbon emission. The addressed problem was permutation flowshop (PFS) scheduling where the carbon emissions & the makespan has to be minimized. All the algorithms are coded in MATLAB® and are executed on a PC with 3.20 Gigahertz frequency and 4GB RAM. Shao et al. (2016) developed a self-guided differential evolution with neighborhood search, called NS-SGDE for solving the problem where ‘n’ jobs to be processed on ‘m’ machine with identical order (PFSSP). The algorithm developed in three stages; Firstly, DHS algorithm integrated with (NEH, Raj, FRB1) was proposed as initial method for the NS-SGDE algorithm. Next, the probabilistic model of EDA was used to generate the guided individual that was applied to guide the DE-based exploration. Some crossover and mutation operators like INSERT, SBOX, SJOX were used to obtain decent solutions. At last, the neighborhood search was constructed which is based on the variable neighborhood search (VNS) technique to improve the capabilities search and discovering optimal result. The Taguchi method of design of experiment was used to analyze the sensitivity of NS-SGDE parameters on its performance. The concurrence of NS-SGDE for PFSSP was analyzed on the bases of the theory of Markov chain. The NS-SGDE was coded in Java (JDK 1.6) language and numerical experiments were performed on a PC with Intel CPU T6600 2.2 GHz Processor and 2.0GB RAM memory. Deng et al. (2017) presented a Competitive memetic algorithm (CMA) for solving the (MODPFSP) with the makespan and total tardiness criteria. Two populations are employed to optimize two different objectives, and the competition among multiple search operators and the knowledge-based local search are performed. Besides, the interaction between the two populations is designed to improve the balance of the two objectives. The influence of the parameters on the performance of the CMA is investigated by using the Taguchi method of design-of-experiment. The CMA was coded in C language and run on PC with Intel i5-3470 processor and 8GB RAM under Windows 7. Laribi et al. (2016) proposed an integer linear model to minimize the makespan for the classical FSS problem where the jobs need additional non-renewable resources for their processing. The modelling was based on the model presented by Carrera (2010) for single machine environment with consumable resource constraint to minimize makespan. A Johnson algorithm was adopted for the two machines flowshop with no renewable resources constraints on the second machine. The results show Johnson algorithm was efficient to solve small problems for one no renewable resource, but when the number of resource increases then results deviates from the optimal solutions. Then the NEH algorithm was proposed to enhance the performance of results on the m machines flowshop under resources constraints. CPLEX v12.2 was used to solve the mathematical model, while all the algorithms were programmed in JAVA language and run on i3 PC with 1.5GHz. Lin et al. (2017) studied the scheduling problem of a premutation flow-shop, which was combined with distributed assembly system known as (DAPFSP) problem. A Backtracking search optimization algorithm (BSA) with Hyper-heuristic approach called as (BS-HH) algorithm was proposed for minimizing makespan in the problem. The low-level heuristics (LLHs) were designed and implemented in the BS-HH for finding its optimal sequences, so that finest solutions can be achieved for the DAPFSP problem. The BS-HH was compared against the state-of-the -art algorithms and the optimality of the solutions were evaluated on the bases of ARPD, Taguchi method and t-test. Core i5-4210U processor with clock speed of 2.40 GHz and 4 GB RAM was used to evaluate experimental results. BS-HH was coded in Visual C++6.0. Ramezanian et al. (2017) proposed two new integrated models, where in both new proposed models, a permutation flow shop scheduling is assumed for the production system and delivery with two difference methods named direct delivery method and delivery with routing decision are assumed for delivery system. The objective is to minimize the total cost for integrated model, which includes sum of the production cost and delivery cost. For solving this Imperialist competitive algorithm (ICA) was selected and some effective policies are introduced and added for improving the original ICA performance. Based on main changes in original ICA steps, it called as Improved Imperialist competitive (I-ICA) algorithm. Taguchi method is also applied for parameter setting of I-ICA to achieve robust results. Based on computational results three important outcomes are observed. First, proposed I-ICA algorithm is efficient algorithm to solve both models. Second, the integration idea for production and delivery operations substantially reduce the total cost of the system. Finally, with integration approach, delivery with routing decision is clearly better than direct delivery method problem. The models are solved by using the optimization software GAMS 23.8 for small size instance and the solver CPLEX 12.4 for medium and large size instances. The algorithm was implemented in MATLAB 7.6.0.324 (R2008a) and run on a PC with 2.80 GHz and 192 GB of RAM memory. Ye et al. (2017) purposed an average idle time (AIT) heuristic for no-wait flow shop production. First of all, the present idle times and future idle times were considered, proposing an initial sequence algorithm, and then the insertion and neigh-borhood exchanging methods are to further improve solutions. The statistical tool ANOVA and paired t-test were used to verify the effectiveness of the AIT heuristic for large-scale instance based on (ARPD). The ARPD, MPD, and PBS are used to evaluate the effectiveness of each heuristic based on small and large-scale instances. The AIT heuristic based on 600 small-scale instances and 120 instances in Taillard's benchmarks, outperforms among the compared heuristics with the same computational complexity. The performance of the AIT heuristic, was equated with the LC, ADT and CH heuristics for solving $F_m|nwt|C_{max}$ problem. All heuristics were framed in MATLAB and run on a Dell Precision T1700 with Intel Core i5-4590 CPUs of 3.3 GHz. Sioud et al. (2017) presented an Enhanced Migrating Birds Optimization (EMBO) Algorithm and a new STH heuristic, for resolving the PFS problem with SDST considering objective of reduce the Makespan. An adapted neighborhood search technique was used to enhanced the migrating bird optimization based on the swap and forward insertion moves. For improving the neighborhood search diversification, a tabu list was presented that is based on the same mechanisms as in tabu search metaheuristics. A circular tabu list containing swap and forward insertions was considered, to avoid premature convergence a restart mechanism was also introduced which is actual a fast way for diversification inside the EMBO. For solving the small, medium and large instances, the STH heuristic is faster than all the other existing heuristics. The results are evaluated using Relative percentage deviation (RPD) and Student t-test, that shows both the algorithms are efficacious and efficient for resolving the problem. The algorithm was structured in C++. Kazemi et al. (2017) explores the Assembly (FSS) problem with two-stages. The first stage consists of m independent machines and in the second stage there are multiple indistinguishable assembly machines for assemble the components. The scheduling problem consists batched delivery system and multi-objectives, where the sum of tardiness and delivery cost has to be minimized. Due to the NP-hard problem, a MIP model was proposed. The MIP model wasn't able to find the solutions of large size problems at reasonable time. Then the Imperialist competitive algorithm (ICA) and the Hybrid imperialist competitive algorithm (HICA) were proposed for solving large-size problems. The (RPE) and Wilcoxon signed-rank test were conducted to equate the performance of proposed algorithms ICA and HICA. During the experiment it was observed that run-time of ICA is less than HICA, but results indicate that the HICA has better performance than the ICA. GAMS 24.1.2 was used to code the (MIP) model and solved by using CPLEX software, whereas the ICA and HICA algorithms were coded in MATLAB R2013a. Both the algorithms and MIP model were executed in a system with intel Pentium B950 0.2.1 GHz processor along with 4 GB of RAM. Laxmi Lalitha et al. (2017) formulated an MILP model for the [N-1] (1) + N(m) hybrid flowshop (HFS) with lot streaming problem to make a schedule that minimizes the makespan. There was one machine in each of the first (N-1) stages and m machines in stage N. The model gives optimal makespan with optimal number of sub-lots, sub-lot sizes, sub-lot sequence, and job sequence. Although the mathematical model performs well, the computational time for large problems is long and optimum solutions become elusive. Hence, an algorithm was proposed. The percentage deviation (PD) of the objective function value of the algorithm is calculated considering the objective function value of the mathematical model as lower bound. The average percentage deviation (APD) for different small size problem sets is also calculated. The results show that the algorithm provides near optimal solution within a very short computational time. The mathematical model and algorithm are coded in LINGO11.0 software package and MATLAB, respectively and are executed in a PC with Intel core i3 processor (2.13 GHz) and 3GB RAM. Samarghandi et al. (2017) developed five mathematical models namely, a MIP model, two quadratic MIP formulations, and two constraint programming models for the NP-hard FSS problem under the no-wait conditions with due date constraints and makespan criterion. A novel graph presentation for the problem was developed, and an Efficient algorithm was proposed for solving the problem. IBM ILOG CPLEX V12.6 was used to resolve the mathematical models and the Algorithms were coded by Microsoft Visual C ++ 2013. The experiments were achieved on Intel Pentium IV PC with 2 GHz processor and 2 GB RAM. Liu et al. (2017) proposed an enhanced HGA for investigating a specific Fuzzy Flowshop scheduling (FFS) problem with a SDST constraint. The aim of the FFS problem is to reduce the energy consumption with respect to minimizing overall makespan and the tardiness between the jobs, this is achieved by determining optimal job sequences and state transition between machines. Largest common pattern (LCP) scheme along with probabilistic heuristics were utilized to boost the evolution performance of the proposed GA algorithm. The GA was superior to solve large and medium sized problems and produce better-quality results with an average performance improvement of 46.5% than the random key GA technique. While setting the parameters of the GA it was observed that there are two critical factors, the maximum setup time and the ratio between setup-to-waiting time affects the tardiness and energy competency of the problem. Both the proposed approach and RKGA approach are coded in C++ and run on a desktop computer with an Intel Core i7 CPU, 4 GB RAM, and a 64-bit Windows 7 operating system. Shao et al. (2017) addressed a no-idle permutation flow shop problem (NIPFSP). A memetic algorithm with hybrid node and edge histogram (MANEH) was presented to resolve the problem with minimize the maximum completion time criterion. A modified accelerate NEH approach combines with hybrid initial method and random initialization to generate favourable solutions. With the implementation of random sample crossover, a hybrid node and edge histogram matrix (NEHM) was developed, The NEHM was built with the major arrangements in population, and the sampling was to generate new sequences for it. Then an upgraded general variable neighborhood search technique with the simulated annealing acceptance (GVNS-SA) was designed, that uses local search in the inner loop and for deciding optimal solutions for the next iterations SA probability was used. The parameters of MANEH were tuned by implementing multi-factor analysis of variance. The computational results indicate the effectiveness of the MANEH algorithm with makespan criterions. All used algorithms were re-coded in Java, and experiments were executed on a server with two Intel Xeon E5-v2620 processors (24 cores) and 64G RAM. Nouri and Ladhari (2017) proposed a multi-objective Genetic algorithm (MBGA) for the bi-objective permutation flowshop scheduling with blocking constraint. The goal of scheduling is find optimal pareto solutions for minimizing the makespan and flow time. NSGA-II technique was used for finding locally Pareto-optimal frontiers for the problem and the NEH heuristic was used for generating initial populations. The MBGA was compared against SPEA-II (Strength Pareto Evolutionary Algorithm). Visual C++ was used to code the algorithms, and run on an Intel Pentium IV 2.4GHz PC with 512 MB of memory.

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 optimize 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 (SDST) 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 employs to the underutilization of resources, results in problem of backlogging too. The problems with backlogging, (SDST) 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 allows 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 streamed 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 keep in mind that optimization of makes-pan 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 aims 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 (FSS) 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), Imperialist competitive algorithm (ICA), Competitive memetic algorithm (CMA), Grey wolf optimizer (GWO), and differential evolution (DE). These heuristics are used alone or can be combined with one another making some 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 initializing 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, CPLEX, GAMS, FORTRAN and languages for coding are java, 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) Every 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 follow 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, two jobs are no to be processed on a single 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 (FSS) 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 (MFSP) 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 (MFSP) problem 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 (FSS) problem related to (SDST) with each part under full backlogging consideration.

(iii) To simulate a novel heuristic algorithm for solving the (MFSP) problem with stochastic parameters by applying Hybrid Genetic Algorithm (HGA) is proposed in the research work.

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

3.5 RESEARCH METHODOLOGY

The formulation study proposes the system architecture algorithm based on the heuristics of sequence based setup time, completion time, tardiness, lateness etc. in environment of multi-objective flow-shop scheduling for backlogging. The proposed research will focus on the mathematical empirical relation framing based on the Genetic Algorithm (GA). In this context, the completion time would be optimum for the effective process performance. The GA can be Hybridized By using other algorithms and searching techniques. This would make the effective utilization of resources such as workers, tools, inventory etc and timely delivery of product specific. Also, the study would comprise of operations specific methods of Branch and Bound Algorithm, Genetic Algorithm, Johnson's Algorithm and Simulated Annealing at last as optimization tool. The proposed research work would develop a novel hybrid algorithm based on the concoction of multi-objective problems of time-and-sequence dependent parameters and consequently, it would sort out the problem of transposition of sequences. The novel hybrid heuristics algorithm would be designed for the elimination of excess completion times/lateness based on process time. The expected research outcome is proposed to be obtained as minimum total processing time or make-span by generating new hybrid Genetic Algorithm, increase the population size and also number of iterations to achieve the feasible optimal results. In this regard, to achieve the minimization of mean flow time, multi-objective functions should be formed according to the time considerations that may affect the process. Then obtained value i.e. the minimum time can be validated using MATLAB. In the preliminary stage of literature review, it is found that the minimum total processing time or make span has not been computed in heuristics. If the mean flow time and vacant time of machines are further determined and their tardiness is computed then the optimization and simulation of tardy jobs and respective jobs is possible. Consequently, the proposed research gaps can justify the hybrid heuristics algorithm.

3.6 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 neigh-bouring 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.

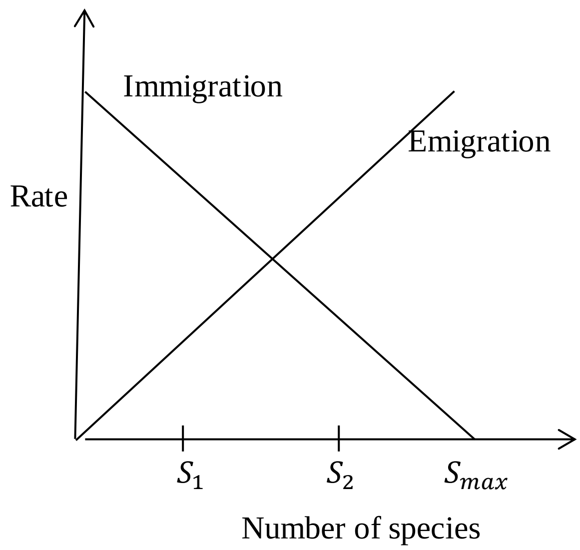

Figure 3.1: Depiction of Relation between Emigrations, Immigration Rates w.r.t. Number of Species for BBO Algorithm

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.

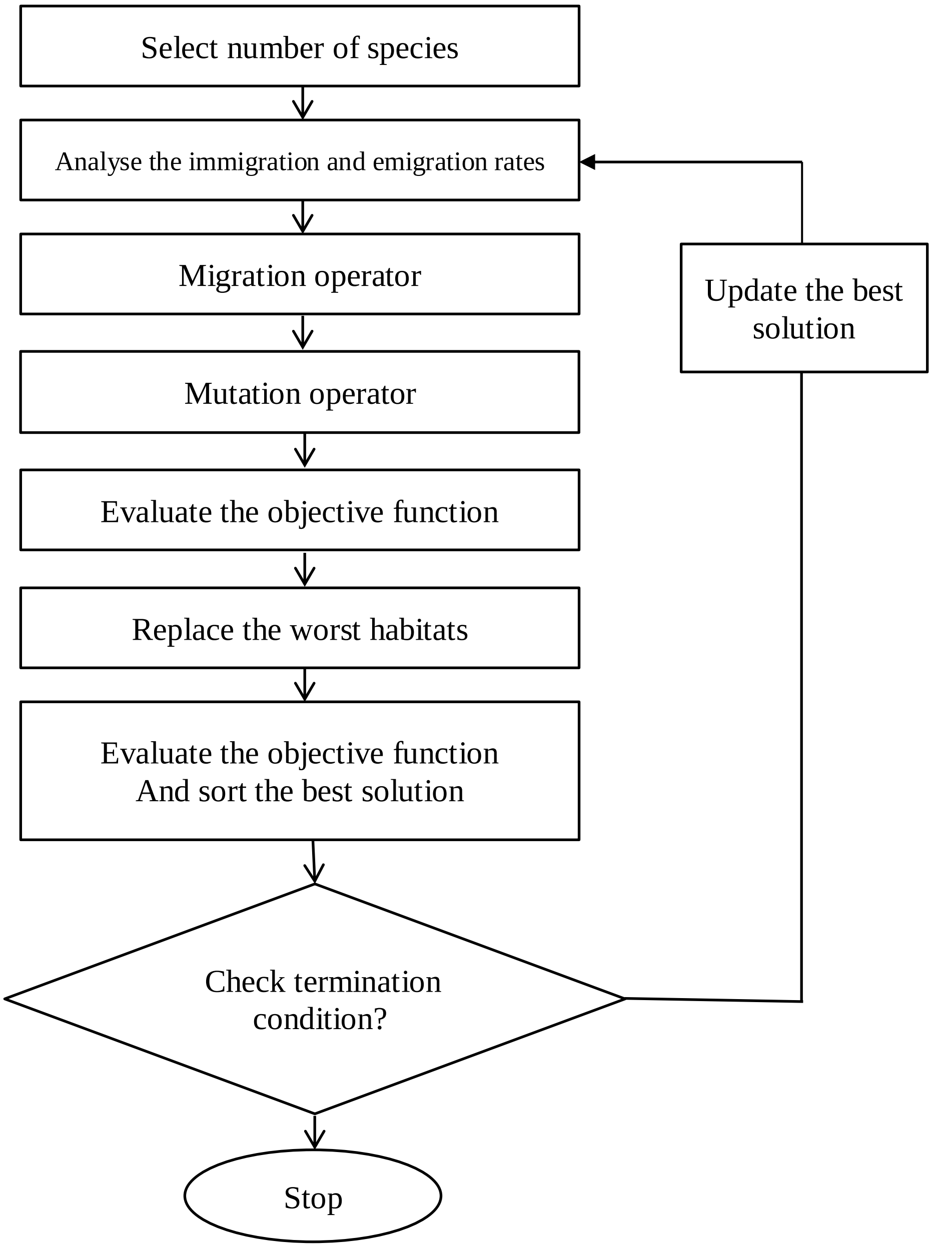

Figure 3.2: Depiction of Various Steps Included in BBO for Optimization of MFSP

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.

3.7 Simulated Annealing (SA)

SA is a descent algorithm modified by random ascent moves in order to escape local minima which are not global minima. The annealing algorithm simulates a nonstationary finite state Markov chain whose state space is the domain of the cost function to be minimized. Importance sampling is the main principle that underlies SA. It has been used in statistical physics to choose sample states of a particle system model to efficiently estimate some physical quantities. Importance sampling probabilistically favours states with lower energies. SA is a general-purpose, serial algorithm for finding a global minimum for a continuous function. When performing SA, theoretically a global minimum is guaranteed to be reached with high probability. The artificial thermal noise is gradually decreased in time. T is a control parameter called computational temperature, which controls the magnitude of the perturbations of the energy function $E(\boldsymbol{x})$. The probability of a state change is determined by the Boltzmann distribution of the energy difference of the two states: $P = e{-}\Delta E/T$. The probability of uphill moves in the energy function ($\Delta E > 0$) is large at high T, and is low at low T. SA allows uphill moves in a controlled fashion: It attempts to improve on greedy local search by occasionally taking a risk and accepting a worse solution.

1. Initialize the system configuration.
   Randomize $\boldsymbol{x}(0)$.
2. Initialize T with a large value.
3. **Repeat**:
   a. **Repeat**:
      i. Apply random perturbations to the state $\boldsymbol{x} = \boldsymbol{x} + \Delta\boldsymbol{x}$.
      ii. Evaluate $\Delta E(\mathbf{x}) = E(\mathbf{x} + \Delta\mathbf{x}) - E(\mathbf{x})$:
         if $\Delta E(\mathbf{x}) < 0$, keep the new state;
         otherwise, accept the new state with probability $P = e^{-\Delta E/T}$

   until the number of accepted transitions is below a threshold level.
   b. Set $T = T - \Delta T$.
   until T is small enough.

Figure 3.3: Steps in Stimulated Annealing

3.8 Imperialist competitive algorithm (ICA)

Imperialist competitive algorithm (ICA) which first was proposed by Atashpaz and Lucas is a population-based effective meta-heuristic used for solving different optimization problems. This algorithm is inspired from social-political behaviors. Each individual of the population in ICA is named a “country” (like chromosome in the GA). Initial population ICA begins with an initial population which is randomly generated. Next, we have to select some of the best countries, having the lowest total cost values, with the size of Nimp from Npop, and set them to be imperialists. Assimilation and revolution all colonies are divided among the imperialist countries and empires are formed which compete among empires and dislocate each other. Then the colonies move toward their relevant imperialists within the cultural state space in each iteration of algorithm. This step is called “assimilation”. Exchanging positions of the imperialist and a colony Once assimilation and revolution operations are performed for an empire, the value of new position of colonies are compared with position of imperialist. If we find any colony that is better than imperialist, then we swap imperialist with that colony.

Begin ICA
Set algorithm parameters
Generate first population.
Sort the first population and select imperialists.
Form the empires.
***While** (the algorithm stopping criterion is not passed) do*
***While** (all empires selected) do Choose empire*
***While** (all colonies selected) do*
Choose colony
Assimilate colony
Revaluate colony
Compare two new costs and substitute colony with new one
End ***while
Descend all colonies of empire.
***If** (Colon y with a lower cost than its imperialist)*
***Then** exchange the position of them.*
Update the location of the empire.
End **While
Find weakest empire based on its total cost.
Send one of weakest empires' colonies to the more powerful empire.
***If** there is an empty empire **Then** Omit the empire and possess its imperialist for the best empire*
End *While
End ICA

Figure 3.4: pseudo code of the ICA algorithm

REFERENCES

1. Johnson, S. M. (1954), "*Optimal Two and Three Stage Production Schedules with Setup Times Included*", Naval Research Logistics, Vol.1, 1, pp. 61–68.
2. Röck, H. (1984), ''Some new results in flow shop scheduling'', Mathematical Methods of Operations Research, 28(1), 1-16.
3. Potts, C. N. (1980). ''An adaptive branching rule for the permutation flow-shop problem'', European Journal of Operational Research, 5(1), 19-25.
4. Liu, G., Luh, P. B., & Resch, R. (1997). ''Scheduling permutation flow shops using the Lagrangian relaxation technique'', Annals of Operations Research, 70, 171-189.
5. Panwalkar, S. S., & Koulamas, C. (2017). ''On the dominance of permutation schedules for some ordered and proportionate flow shop problems'', Computers &Industrial Engineering, 107, 105-108.
6. Karabati, S., Kouvelis, P., & Kiran, A. S. (1992). ''Games, critical paths and assignment problems in permutation flow shops and cyclic scheduling flow line environments'', Journal of the Operational Research Society, 241-258.
7. Deng, J., & Wang, L. (2017). ''A competitive memetic algorithm for multi-objective distributed permutation flow shop scheduling problem'', Swarm and Evolutionary Computation, 32, 121-131.
8. Samarghandi, H., & Behroozi, M. (2017). ''On the exact solution of the no-wait flow shop problem with due date constraints'', Computers & Operations Research, 81, 141-159.
9. Said Ashour (1970)," A branch-and-bound algorithm for flow shop scheduling problems", AIIE Transactions, 2(2), 172-176.
10. Reddi, S. S., & Ramamoorthy, C. V. (1972). ''On the flow-shop sequencing problem with no wait in process'', Operational Research Quarterly, 323-331.
11. Grabowski, J., Skubalska, E., & Smutnicki, C. (1983). ''On flow shop scheduling with release and due dates to minimize maximum lateness'', Journal of the Operational Research Society, 34(7), 615-620.
12. Szwarc, W. (1988). ''The clustered flow-shop problem'', Mathematical Methods of Operations Research, 32(5), 315-322.
13. Panwalkar, S. S. (1991). ''Scheduling of a two-machine flowshop with travel time between machines'', Journal of the operational Research Society, 609-613.

14. Liao, C. J. (1993). ‘’Minimizing the number of machine idle intervals with minimum makespan in a flow-shop’’, Journal of the Operational Research Society, 817-824.
15. Riezebos, J., Gaalman, G. J., & Gupta, J. N. (1995). ‘’Flow shop scheduling with multiple operations and time lags’’, Journal of intelligent manufacturing, 6(2), 105-115.
16. Ronconi, D. P., & Armentano, V. A. (2001). ‘’Lower bounding schemes for flowshops with blocking in-process’’, Journal of the Operational Research Society, 52(11), 1289-1297.
17. Pan, J. H., & Chen, J. S. (2003). ‘’Minimizing makespan in re-entrant permutation flow-shops’’, Journal of the Operational
18. Yanai, S., & Fujie, T. (2006). ‘’A three-machine permutation flow-shop problem with minimum makespan on the second machine’’, Journal of the Operational Research Society, 57(4), 460-468.
19. Pan, Q. K., & Wang, L. (2008). ‘’No-idle permutation flow shop scheduling based on a hybrid discrete particle swarm optimization algorithm’’, The International Journal of Advanced Manufacturing Technology, 39(7), 796-807.
20. Ahmadizar, F., & Farahani, M. H. (2012). ‘’A novel hybrid genetic algorithm for the open shop scheduling problem’’, The International Journal of Advanced Manufacturing Technology, 62(5-8), 775-787.
21. Costa, A., Cappadonna, F. A., & Fichera, S. (2014). ‘’A novel genetic algorithm for the hybrid flow shop scheduling with parallel batching and eligibility constraints’’, The International Journal of Advanced Manufacturing Technology, 75(5-8), 833-847.
22. Fung, J., Zinder, Y., & Singh, G. (2016). ‘’Flexible Flow Shop with Storage: Complexity and Optimisation Methods’’, IFAC-PapersOnLine, 49(12), 237-242.
23. Kampmeyer, M., Knust, S., & Waldherr, S. (2016). ‘’Solution algorithms for synchronous flow shop problems with two dominating machines’’, Computers & Operations Research, 74, 42-52.
24. Liepins, G. E., & Hilliard, M. R. (1989). ‘’Genetic algorithms: Foundations and applications’’, Annals of operations research, 21(1), 31-57.
25. Mirabi, M. (2014). ‘’A novel hybrid genetic algorithm to solve the sequence-dependent permutation flow-shop scheduling problem’’, International Journal of Advanced Manufacturing Technology, 71.

26. Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). ‘‘Equation of state calculations by fast computing machines’’ The journal of chemical physics, 21(6), 1087-1092.
27. Low, C., Yeh, J. Y., & Huang, K. I. (2004). ‘‘A robust simulated annealing heuristic for flow shop scheduling problems’’, The International Journal of Advanced Manufacturing Technology, 23(9-10), 762-767.
28. Land, A. H., & Doig, A. G. (1960). ‘‘An automatic method of solving discrete programming problems’’, Econometrica: Journal of the Econometric Society, 497-520.
29. McMahon, G. B., & Burton, P. G. (1967). ‘‘Flow-shop scheduling with the branch-and-bound method’’, Operations Research, 15(3), 473-481.
30. Dorigo M. and Gambardella, L. M. (1997), “Ant Colonies for the Travelling Salesman Problem”, Biosystems, Vol. 43, 2, pp. 73-81.
31. Dorigo, M., & Gambardella, L. M. (1997). ‘‘Ant colony system: a cooperative learning approach to the traveling salesman problem’’, IEEE Transactions on evolutionary computation, 1(1), 53-66.
32. Yagmahan, B., & Yenisey, M. M. (2010). ‘‘A multi-objective ant colony system algorithm for flow shop scheduling problem’’. Expert Systems with Applications, 37(2), 1361-1368.
33. Eberhart, R., & Kennedy, J. (1995, October). ‘‘A new optimizer using particle swarm theory’’, In Micro Machine and Human Science, 1995. MHS’95., Proceedings of the Sixth International Symposium on (pp. 39-43). IEEE.
34. Liao, C. J., Tjandradjaja, E., & Chung, T. P. (2012). ‘‘An approach using particle swarm optimization and bottleneck heuristic to solve hybrid flow shop scheduling problem’’, Applied Soft Computing, 12(6), 1755-1764.
35. Geng, J. C., Cui, Z., & Gu, X. S. (2016). ‘‘Scatter search based particle swarm optimization algorithm for earliness/tardiness flowshop scheduling with uncertainty’’, International Journal of Automation and Computing, 13(3), 285-295.
36. Glover, F. (1986), “Future Paths for Integer Programming and Links to Artificial Intelligence”, Computers and Operations Research, Vol. 13, 5, pp. 533-549.
37. Glover, F. (1989), “Tabu Search-Part 1”, ORSA Journal on Computing, Vol. 1, 2, pp. 190-206.

38. Yip, Y., Cheng, C. Y., & Low, C. (2006). ‘‘Sequencing of an M machine flow shop with setup, processing and removal times separated’’, The International Journal of Advanced Manufacturing Technology, 30(3), 286-296.
39. Storn, R., & Price, K. (1997). ‘‘Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces’’, Journal of global optimization, 11(4), 341-359.
40. Deng, G., & Gu, X. (2012). ‘‘A hybrid discrete differential evolution algorithm for the no-idle permutation flow shop scheduling problem with makespan criterion’’ Computers & Operations Research, 39(9), 2152-2160.
41. Zhou, B. H., Hu, L. M., & Zhong, Z. Y. (2016). ‘‘A hybrid differential evolution algorithm with estimation of distribution algorithm for reentrant hybrid flow shop scheduling problem’’, Neural Computing and Applications, 1-17.
42. Farmer, J. D., Packard, N. and Perelson, A. (1986), “The Immune System, Adaptation and Machine Learning”, Physica D, Vol. 2, pp. 187-204.
43. Alisantoso, D., Khoo, L. P., & Jiang, P. Y. (2003). ‘‘An immune algorithm approach to the scheduling of a flexible PCB flow shop’’, The International Journal of Advanced Manufacturing Technology, 22(11-12), 819-827.
44. 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
45. Shao, W., Pi, D., & Shao, Z. (2016). ‘‘A hybrid discrete optimization algorithm based on teaching-probabilistic learning mechanism for no-wait flow shop scheduling’’, Knowledge-Based Systems, 107, 219-234.
46. Simon, D. (2008), “Biogeography-Based Optimization”, IEEE Transactions on Evolutionary Computation, Vol. 12, 6, pp. 702-713.
47. Lin, J., & Zhang, S. (2016). ‘‘An effective hybrid biogeography-based optimization algorithm for the distributed assembly permutation flow-shop scheduling problem’’, Computers & Industrial Engineering, 97, 128-136.
48. Torkashvand, M., Naderi, B., & Hosseini, S. A. (2017). ‘‘Modelling and scheduling multi-objective flow shop problems with interfering jobs’’, Applied Soft Computing, 54, 221-228.
49. Ding, J. Y., Song, S., & Wu, C. (2016). ‘‘Carbon-efficient scheduling of flow shops by multi-objective optimization’’, European Journal of Operational Research, 248(3), 758-771.

50. Han, Y., Gong, D., Jin, Y., & Pan, Q. K. (2016). ‘‘Evolutionary multi-objective blocking lot-streaming flow shop scheduling with interval processing time’’, Applied Soft Computing, 42, 229-245.
51. Szwarc, W., & Liu, J. J. (1989). ‘‘An approximate solution of the flow-shop problem with sequence dependent setup times’’, Mathematical Methods of Operations Research, 33(6), 439-451.
52. Nagano, M. S., Da Silva, A. A., & Lorena, L. A. N. (2014). ‘‘An evolutionary clustering search for the no-wait flow shop problem with sequence dependent setup times’’, Expert Systems with Applications, 41(8), 3628-3633.
53. Sioud, A., & Gagné, C. (2017). ‘‘Enhanced Migrating Birds Optimization Algorithm for the Permutation Flow Shop Problem with Sequence Dependent Setup Times’’, European Journal of Operational Research.
54. Babaei, M., Mohammadi, M., & Ghomi, S. F. (2014). ‘‘A genetic algorithm for the simultaneous lot sizing and scheduling problem in capacitated flow shop with complex setups and backlogging’’, The International Journal of Advanced Manufacturing Technology, 70(1-4), 125-134.
55. Ziaeifar, A., Tavakkoli-Moghaddam, R., & Pichka, K. (2012). ‘‘Solving a new mathematical model for a hybrid flow shop scheduling problem with a processor assignment by a genetic algorithm’’, The International Journal of Advanced Manufacturing Technology, 61(1), 339-349.
56. Fattahi, P., Hosseini, S. M. H., & Jolai, F. (2013). ‘‘A mathematical model and extension algorithm for assembly flexible flow shop scheduling problem’’, The International Journal of Advanced Manufacturing Technology, 1-16.
57. Mousavi, S. M., Mousakhani, M., & Zandieh, M. (2013). ‘‘Bi-objective hybrid flow shop scheduling: a new local search’’, The International Journal of Advanced Manufacturing Technology, 1-18.
58. Mirabi, M. (2014). ‘‘A novel hybrid genetic algorithm to solve the sequence-dependent permutation flow-shop scheduling problem’’, International Journal of Advanced Manufacturing Technology, 71.
59. Komaki, G. M., & Kayvanfar, V. (2015). ‘‘Grey Wolf Optimizer algorithm for the two-stage assembly flow shop scheduling problem with release time’’, Journal of Computational Science, 8, 109-120.

60. Xiao, Y., Yuan, Y., Zhang, R. Q., & Konak, A. (2015). ‘‘Non-permutation flow shop scheduling with order acceptance and weighted tardiness’’, Applied Mathematics and Computation, 270, 312-333.
61. Ferrer, A., Guimarans, D., Ramalhinho, H., & Juan, A. A. (2016). ‘‘A BRILS metaheuristic for non-smooth flow-shop problems with failure-risk costs’’, Expert Systems with Applications, 44, 177-186.
62. Shao, W., & Pi, D. (2016). ‘‘A self-guided differential evolution with neighborhood search for permutation flow shop scheduling’’, Expert Systems with Applications, 51, 161-176.
63. Deng, J., & Wang, L. (2017). ‘‘A competitive memetic algorithm for multi-objective distributed permutation flow shop scheduling problem’’, Swarm and Evolutionary Computation, 32, 121-131.
64. Laribi, I., Yalaoui, F., Belkaid, F., & Sari, Z. (2016). ‘‘Heuristics for solving flow shop scheduling problem under resources constraints’’, IFAC-Papers OnLine, 49(12), 1478-1483.
65. Ramezanian, R., Mohammadi, S., & Cheraghalikhani, A. (2017). ‘‘Toward an integrated modeling approach for production and delivery operations in flow shop system: Trade-off between direct and routing delivery methods’’, Journal of Manufacturing Systems, 44, 79-92.
66. Ye, H., Li, W., & Abedini, A. (2017). ‘‘An improved heuristic for no-wait flow shop to minimize makespan’’, Journal of Manufacturing Systems.
67. Kazemi, H., Mazdeh, M. M., & Rostami, M. (2017). ‘‘The two-stage assembly flow-shop scheduling problem with batching and delivery’’, Engineering Applications of Artificial Intelligence, 63, 98-107.
68. Lalitha, J. L., Mohan, N., & Pillai, V. M. (2017). ‘‘Lot streaming in [N-1] (1)+ N (m) hybrid flow shop’’, Journal of Manufacturing Systems, 44, 12-21.
69. Samarghandi, H., & Behroozi, M. (2017). ‘‘On the exact solution of the no-wait flow shop problem with due date constraints’’, Computers & Operations Research, 81, 141-159.
70. Liu, G. S., Zhou, Y., & Yang, H. D. (2017). ‘‘Minimizing energy consumption and tardiness penalty for fuzzy flow shop scheduling with state-dependent setup time’’, Journal of Cleaner Production, 147, 470-484.

71. Shao, W., Pi, D., & Shao, Z. (2017). ‘‘Memetic algorithm with node and edge histogram for no-idle flow shop scheduling problem to minimize the makespan criterion’’, Applied Soft Computing, 54, 164-182.
72. Nouri, N., & Ladhari, T. (2017). ‘‘Evolutionary multi-objective optimization for the multi-machine flow shop scheduling problem under blocking’’, Annals of Operations Research, 1-18.
73. Taillard, E. (1993). ‘‘Benchmarks for basic scheduling problems’’, European journal of operational research, 64(2), 278-285.
74. Tavakkoli-Moghaddam, R., Rahimi-Vahed, A. R., & Mirzaei, A. H. (2008). ‘‘Solving a multi-objective no-wait flow shop scheduling problem with an immune algorithm’’, The International Journal of Advanced Manufacturing Technology, 36(9-10), 969-981.
75. Wu, T., Shi, L., Geunes, J., & Akartunalı, K. (2011). An optimization framework for solving capacitated multi-level lot-sizing problems with backlogging. European Journal of Operational Research, 214(2), 428-441.