Job shop scheduling problem using hybrid ant colony optimization and genetic algorithm

thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science and Engineering

Submitted By
Arti Patel
(801632003)

Under the supervision of:
Dr. Vinay Arora
Assistant Professor, CSED

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY
PATIALA – 147004

June 2018

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*Job shop scheduling problem using hybrid ant colony optimization and genetic algorithm*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* submitted in Computer Science and Engineering Department of Thapar Institute of Engineering and Technology, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Vinay Arora* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University. In case of any discrepancy(even after publication of thesis),I (Arti Patel) take the full responsibility.

Signature:

(Arti Patel)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge. In case of any dispute or discrepancy (even found after submission of the thesis),only candidate will be responsible.

(Dr. Vinay Arora)

AssistantProfessor,

CSED,TIET

Patiala

Scanned by CamScanner

Acknowledgement

First of all, I would like to express my gratitude towards **Thapar Institute of Engineering and Technology**, for pro-viding me a platform to do my thesis work at such an esteemed institute.

I wish to express my respect, deep sense of gratitude and indebtedness to my guide **Dr. Vinay Arora**, Assistant Professor, Computer Science and Engineering Department,Thapar University, Patiala for their invaluable and enthusiastic guidance, useful suggestions, unfailing patience and sustained encouragement throughout this work.

I would like to thank **Dr. Maninder Singh**, Head of Computer Science and Engineering Department, Thapar University, Patiala for kind help, guidance, encouragement and providing the necessary facilities to carry out my research. I am indebted to the faculty members of the department for valuable suggestions, friendly support and full cooperation rendered by all of them.

Any vote of thanks is not enough to thank the supreme power "**The GOD**" one who has always guided me to work on the right path of the life. Without his grace, this would never come to be today's reality. With special thanks, I dedicate this thesis to GOD.

Last but not the least I would like to thank my **family** and my friends specially **Darshan Patel, Sugandha Budhiraja,** and **Shruti Arora** for their help and support throughout the thesis.

Arti Patel

Abstract

Job-shop scheduling techniques play a significant role in various parallel applications. An efficient job-shop scheduling technique not only provides high availability of resources to users but also enhances the performance of parallel machines. Job-shop scheduling techniques are a typical NP-hard problem. Currently, numerous researchers have solved job-shop scheduling problems by considering the well-known meta-heuristic techniques. These techniques are Genetic algorithm (GA), Particle swarm optimization (PSO), Variable neighborhood search (VNS), Ant colony optimization (ACO), BAT algorithm (BA), Artificial bee colony (ABC) *etc*. However, these techniques suffer from one of these issues: premature convergence, poor convergence speed, initially selected random solutions and stuck in local optima.

To handle the issues associated with existing meta-heuristics based job-shop scheduling techniques, in this research work, hybrid scheduling techniques are designed. Hybridization of meta-heuristic based job-shop scheduling techniques is achieved by integrating the ACO with GA. It has an ability to overcome several issues associated with existing techniques such as premature convergence, poor convergence speed, initially selected random solutions and stuck in local optima issues.

To attain the objectives of this research work, a step by step methodology has been used. In the proposed techniques, ACO is used to generate the population for GA. Then, the developed solutions are followed by GA, to optimize these solutions by utilizing the mutation and crossover operators. Therefore, the proposed technique has an ability to find the more efficient schedules compared to available meta-heuristic based job-shop scheduling techniques.

The proposed techniques have been compared with other scheduling techniques in terms of makespan, average execution time, and energy consumption. By reducing consumed energy, all the proposed techniques, indirectly reduce carbon emissions and cooling requirements of the parallel machines leading to a further drop in the energy demand and helping in achieving green computing environment.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Outline

This Chapter describes the role of job shop scheduling in distributed environment. This chapter also discusses various models, types and architectures of machines computing. At the end of this chapter, some well-known meta-heuristic techniques are also discussed. These meta-heuristics techniques have been extensively used by the research community to schedule jobs between available set of machines.

1.1 Overview

The job-shop scheduling problem (JSP) plays a significant role in Flexible Manufacturing System (FMS). In JSP, several types of flexibilities are considered, including non-linear alternative process plans of products, alternative machines of operations, and variable processing sequences, such that the performance of the schedule can be potentially improved [10]. Majority of the relative studies was prepared for a static environment, where all manufacturing resources are assumed rightly available and unlimited at all times. In the real-world, however, it is quite common for manufacturing systems to encounter unexpected disruptions such as machine breakdown and rush orders. In such circumstances, the predetermined schedules will lose their optimality or even become infeasible to be executed [11].

To cope with the uncertainties, dynamic scheduling is an alternative which uses decentralized control systems to dispatch jobs when necessary using the information available right at the dispatching time, making a decision by dispatching rules, or other heuristics, rather than generating a full schedule before actual production [12]. Another well-known method is reactive scheduling, or rescheduling, whereby modifications will

be made on the predetermined schedules in face of the disruptions to permit the actual execution. Rescheduling can be conducted either to rearrange the affected operations, which is known as partial rescheduling, or to generate an entirely new schedule for all operations afterward, which is known as total rescheduling [13]. The timing policy is classified into periodical and event-driven: the former defines a time interval between rescheduling actions, during which rescheduling actions are not permitted, while for the latter, the rescheduling action will be performed upon the recognition of any disruption.

1.2 Job shop scheduling

Scheduling can manage several jobs and assignment of resources in a period. A primary objective of scheduling is the assignment of jobs to the available set of machines so that execution time can be minimized. Another feature of scheduling is decision-making process and is mostly used in service and manufacturing organizations. Scheduling takes place by taking help of job-shop scheduling algorithms [14]. Job-shop is a term related with scheduling used at an application level and is a script or program to execute a specific set of jobs. As per the literature survey [15] job shop scheduling is defined as follows:

i. A job-shop scheduling algorithm is a technique in which jobs are optimistically assigned to data center resources.

ii. There is no completely perfect scheduling mechanism available due to different scheduling objectives.

iii. Scheduling algorithms can be executed or implemented under suitable conditions according to assigned applications by a good scheduler.

iv. Scheduling algorithm is a mechanism that solves a problem in seconds, minutes or even hours.

v. Time used for execution of particular algorithm measures the efficiency of that algorithm and so time complexity can be measured from the efficiency.

Time complexity plays a significant role in the execution of an algorithm. There are some time complexity algorithms used for job-shop execution. The problem is feasible, tractable, fast, and efficient which is related to polynomial time algorithm [16].

Class P: It deals with problems having a decision-making process to solve deterministic turing machine in polynomial time. Class P problem can be solved easily and decided

quickly using polynomial time mechanism [16].

Job-shop scheduling problem can be solved easily by matching the jobs to various sets of resources. It is denoted by the triplet (J, S, O). Here, J stands for a set of jobs, S shows the feasible set solutions and O is objective of the problem. Further scheduling problem can be categorized into two sections, i). Optimization problem and ii). Objective O based decision-related problem.

Firstly, to optimize the problem, it is necessary to search the best-optimized solution from feasible set S. In the case of decision-making problems, searching for an optimized solution is quite easy as compared to other solutions. To achieve the objective, there must be a positive or negative solution to solve the problem from the particularly feasible solution set s which is a subset of set S. So, a decision problem is best as compared to the optimization problem. Scheduling problems find the optimal set of jobs to various resources and it is called combinational optimization problems. It is a category of natural problem and working becomes easy as to solve polynomial algorithms or enumerations. Secondly, if problem solves with decision-making, then it is in the class of NP-complete and has optimized solution, otherwise, it is in NP-hard class [17].

Scheduling can be described as deciding process primarily of this particular allocation of operations on machines ordinary manner the mandatory performance measures including makespan, tardiness, earliness, and flow time can be called. The scheduling procedures on the market may be realistic but will also possess fast solution generating capability. It finds widespread applications inside the manufacturing and service industries. For those deciding process scheduling and sequencing exist at several levels according to most researchers. Those levels are listed below: potential planning, middle-term planning, cash advance planning, predictive planning and reactive/control planning. Sequencing and scheduling environments are classified into four types. Influenced by requirements machine and volume of jobs and machines and the types of assignments, the sequencing, and scheduling environments are classified the subsequent: single machine shop, flow shop, job shop, assembly job shop, hybrid job shop, hybrid assembly job shop, open shop, closed shop.

1.2.1 Single machine scheduling problem

A single machine shop could be broadly classified into two major types:

i. with single processors

ii. with parallel processors

The classification for the only machine scheduling environment is shown in Figure 1.1.

Single machine scheduling with single processor involves single machine to process n jobs. The intention of this concern is to optimize the given performance measure by scheduling n jobs on the only machine. These jobs may well be independent or dependent, to match the condition. As soon as set-up times from the efforts are considered to be independent of the process sequence from the jobs, the scheduling concern is termed as the only machine scheduling issue with independent jobs; otherwise, it truly is termed as single machine scheduling issue with dependent jobs. All the performance measures of the only machine scheduling issue with independent jobs are as follows:

- Minimizing the mean flow time
- Minimizing the utmost lateness
- Minimizing the full tardiness
- Minimizing may be tardy jobs

1.2.2 Other scheduling environments

i. In flow-shop scheduling problems, jobs require m different machines for processing the jobs. Most of the jobs have the process sequence.

ii. In job-shop scheduling problems, n jobs should be processed on m different machines. Each job has process sequence and is also different from one another.

iii. In assembly job-shop, a job-shop with jobs has at least two component items and at least one assembly operation.

iv. In hybrid job-shop, the precedence ordering from the operations of some jobs often is the same.

v. Hybrid assembly job shop combines the features of the assembly and hybrid job-shop.

vi. In open shop scheduling there isn't a process sequence for every job hence the operation from the jobs can be carried out in any order.

vii. Closed shop: this is a job-shop; however all production orders are generated on account of inventory replenishment decisions. To paraphrase, the production isn't afflicted with the customer's order.

Because parallel machine set-up will be displayed for a house block of all these complex environments, choosing a resolution methodology into parallel machine scheduling problem will donor foundation of the best methodologies of the complex production environment. Obtained in this research work, the focus is reached on identical parallel machine scheduling.

Job sequencing is the best way to be made which has a finite selection of service facilities. Scheduling is often used as the allocation of resources eventually to attempt an accumulation of tasks. The practical problem of allocating resources eventually to attempt an accumulation of tasks arises in lots of situations. Typically, however, scheduling will not become significant until some fundamental planning troubles are resolved, this also is necessary to be recognized that scheduling decisions are secondary importance to somewhat of a broader massive amount managerial decisions. Because of example, in manufacturing applications, the select a mental managerial questions involve getting a service come to be manufactured and determining the length of production. Aftermarket studies and economic analyses understand or know to resolve such issues, technological planning concentrates on the question of how the goods should certainly be manufactured. Only after these planning questions are typically answered, and producing resources could very well be known, would n't it become appropriate to keep in mind problems of scheduling. As the second example, the select a mental managerial question in the delivery of healthcare required service may just be offered. Then technological planning manages questions of facility design, equipment utilization, and personnel deployment. Once these decisions have given a profile with the resources available, it is then possible to manage scheduling problems. These examples indicate how to select a mental managerial decisions address themselves to three varieties of questions :

i. What products or services can be provided?

ii. The exact amount scale might it be possible provided? and

iii. What resources are going to be presented?

Determining answers to these questions is the look function, as opposed, the scheduling function presumes that answers to these questions already exist. Therefore, the function of scheduling becomes relevant in a situation where the tasks to remain scheduled continues to be described together with the configuration of resource available continues to be determined.

Two different feasibility constraints are typically in scheduling problems. First, there are actually limits on the capability of available resources. Second, there are actually technological restrictions in the order of which tasks are performed. A remedy to somewhat of a scheduling concern is any feasible resolution these 2 kinds of constraints, so that "solving" a scheduling problem amounts to answering two different questions:

i. Which resources can be allocated that are performing each task?

ii. When will each task be performed?

To paraphrase, the essences of scheduling problems give rise to (1) allocation decisions and (2) sequencing decisions. Scheduling is classified into the following three categories.

i. Single Machine Scheduling (SMS)

ii. Flow Shop Scheduling (FSS)

iii. Job Shop Scheduling (JSS)

1.2.3 Single machine scheduling

When the whole production of any item takes place on a single machine the case of single machine scheduling arises in sequencing problem, as the total time given for complete process is fixed which is equal to the total time required for processing of the machine. In single machine scheduling, the priority is given to the jobs having least processing time should be processed first ie.LPT rule is used. By this, we can reduce the total waiting time and total tardiness of jobs. For eg.Flour mill, the number of programs waiting for running on computer *etc*. Available single machine concern is seen as these conditions.

i. A regular of n independent, single-operation jobs can be found for processing at time zero.

ii. Setup times for one's tasks are independent of job sequence that will be within processing times.

iii. Job descriptors are known in advance.

iv. One machine is continuously available and is also never kept idle while tasks are waiting.

v. Once processing begins even on a job, it will be processed to completion without interruption.

1.2.4 Flow shop scheduling

The main objective of scheduling is to achieve the maximum output in minimum time, within given due date, and to fulfill the requirements of customers up to the mark.For satisfying the desired target, n-jobs are processed on M machines with the assumptions that each machine is allowed to execute only one job at a time, once a job start processing it should be completed and all jobs should be of equal importance and all jobs are independent of each other. Each machine is provided with sufficient waiting time.

Every job takes fixed time for completion which includes setup time, processing time, transporting time, returning time, loading time, unloading time, slack time, delay time *etc*. A heuristic approach is used for finding the optimal solution. A store contains m different machine, and every job consists of w operations, as both versions require a better machine. The flow shop is characterized by a flow of work that's unidirectional. The flow shop issue will be characterized as given below.

i. A few multiple-operation jobs can be found for processing at time zero. (Each job requires m operations and every operation requires a better machine)

ii. Setup times for one's operations are sequence-independent and are within processing times.

iii. Job descriptors are known in advance.

iv. M different machines are continuously available.

v. Individual operations are not primitive.

1.2.5 Job shop scheduling

The classical job shop problem is different from the flow shop injury in one important respect: the flow of work is simply not unidirectional. The aspects of a predicament are a few machines and an accumulation of jobs to remain scheduled. Each job consists of several operations with linear precedence structure as with the flow shop model. Although it is possible to allow numerous operations in the job, the most typical formulation of the store problem specifies that every job has exactly m operations, one on each machine.

1.3 Metaheuristic based job-shop scheduling techniques

A meta-heuristic is a procedure that helps to find the optimal solutions. Meta-heuristics are of various types based on search strategy, single solution based, nature-inspired *etc*.

Evolutionary Computation (EC) is a problem-solving computer-based that system deals with evolutionary processes, In other words, it deals with natural selection. It is the computational model used for the survival of the fittest and reproduction. Evolution through natural selection is selected population of individuals randomly and is a search procedure using the space of possible chromosome values. Overall procedure can be done by using an Evolutionary Algorithm (EA) and is a subset of evolutionary computation. Stochastic search was applied to a given problem so that optimized results

can be produced. It is also a generic population-based optimization algorithm. These algorithms are inspired by biological evolution having four approximate solutions passes through reproduction, recombination or crossover, mutation and selection [18].

Swarm Intelligence (SI) is a most popular approach of Artificial Intelligence (AI) in information technology and biology field. It is applied for obtaining the aggregate activities of natural social swarms like Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Artificial Bee Colony (ABC) and Variable Neighborhood Search (VNS). Swarm individuals or insects are agents have unsophisticated limited capabilities. These agents cooperatively get jobs for their survival by interacting with each other under some behavioral model [19]. Swarm individuals socially interact directly or indirectly to get the optimized solution. Due to incredible efficiency of natural swarm system, researchers started working on the behaviors of social insects and it is fruitful [20].

Evolution strategies For attaining the optimal solution for the numerical problems, evolution strategies are developed. To determine the next generation, the newly produced individual is evaluated and selected by applying different selection schemes. The fitness value is considered while selecting the individuals. Following are the major evolution strategy steps:

i. Population initialization

ii. Repeat

iii. Computing fitness function

iv. Recombine the generation

v. Process of mutation

vi. Evaluation of the new generation

vii. Selection of best generation

viii. until conditions are fulfilled

I. Genetic Algorithm

A Genetic Algorithm (GA) is inspired from biological evolution metaphor of original Darwinian theory and mimics the operation of genetic evaluation by using heuristic search. It is a stochastic search evolutionary algorithm used in the artificial intelligence field of computer science. It optimized the useful solutions and solved all search problems heuristically. It solves each optimization problem that can be explained by chromosome encoding with multiple solutions. GA is independent on multi-dimensional error surface, non-continuous, non-differential and non-parametrical problems. This mechanism understands and explains the concept very quickly without any knowledge of mathematics. Simulations and models are easily transferred with this technique [21].

The fitness functions having poor results generate and block bad chromosome instead of blockage of good chromosome using crossover operation. This will optimize the solution and such optimization problems are known as variant problems. When the population is large that is having complicated subjects, then it's hard to get global optima by GA [15].

GA cannot guarantee constant optimal response time as compared to other artificial intelligence techniques. Due to the probability limit of GA, real-time applications can be optimized by GA. It improves the whole population and is not defined as an individual within this population. In a real system with a simulation model, on-line controlling without testing first becomes unreasonable by using GA [16].

GA is an optimization method based on population and Darwin's theory of evolution [1]. In GA, each possible solution is represented by a chromosome. An initial population is taken randomly which is generated by a random generator and it is used as a starting point. A fitness function is calculated for each chromosome so that it finds whether the chromosome is suitable or not. Crossover and mutation functions are performed on the selected chromosomes and offspring are so that a fresh population is created. This activity is repeated until enough offspring are created [1]. A fitness value associated with each string is a good measurement of the solution represented in this mechanism. Crossover operator randomly selects pairs of strings and output as a set of new pairs. Actually, Crossover rate is responsible for the number of crossover operations. The bits of the string is randomly mutated by the mutation operator. Mutation rate determines the number of mutation operations. Every step in this technique produces a set of a new generation as solutions. Figure 1.1 shows the step-by-step detail of the genetic algorithm.

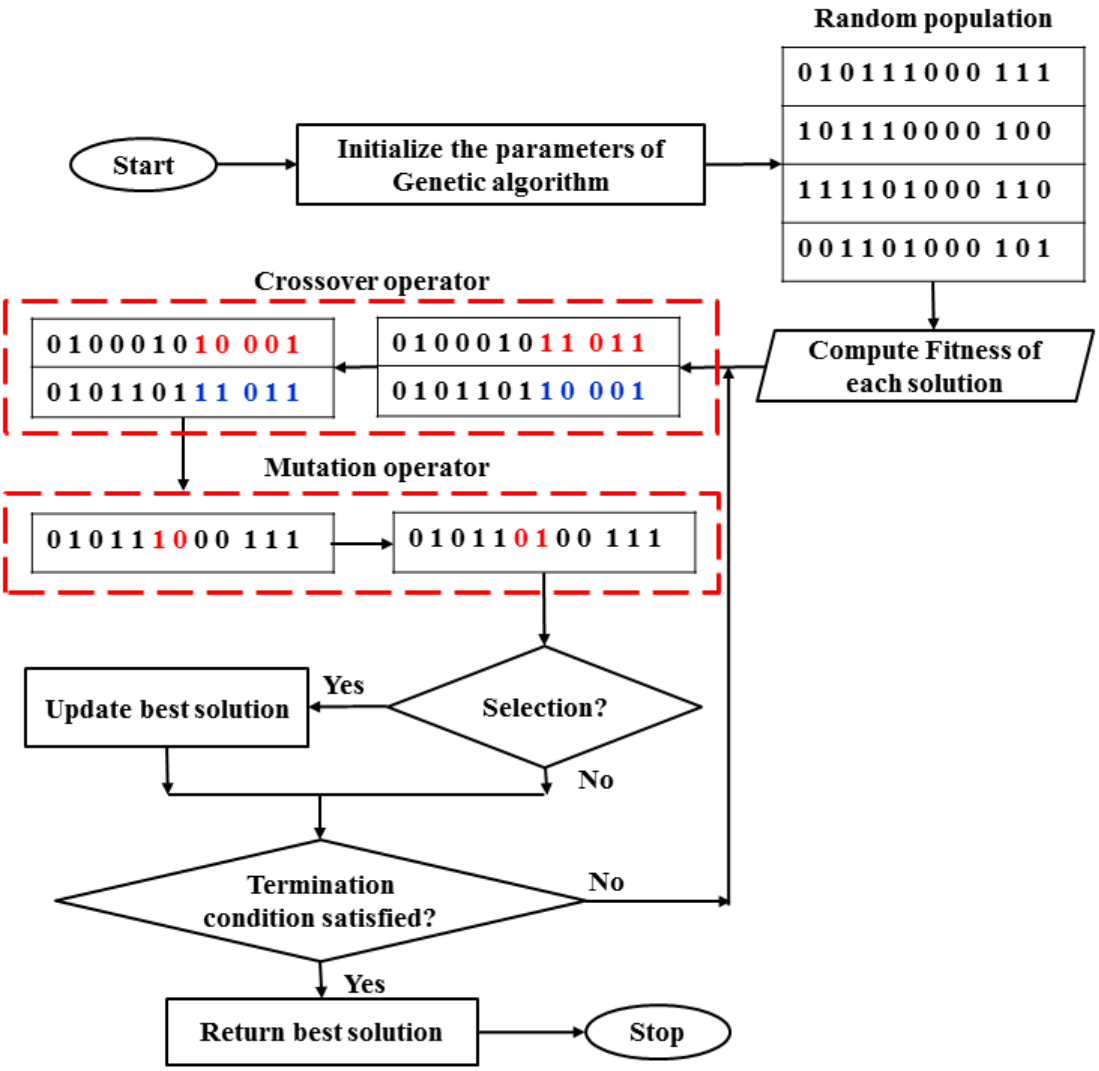

Figure 1.1: A flow of GA process[1]

1. Pick an initial arbitrary population of individuals
2. Evaluate the fitness function of the individuals
3. Repeat until the best individual found
4. The best individuals should be selected
5. Generate new individuals by applying crossover and mutation operators
6. The fitness function for new individuals should be examined
7. The worst individuals are replaced with the best ones
8. until a stopping criterion is met

II. Particle Swarm Optimization (PSO)

PSO is optimized swarmed intelligence stochastic population-based approach inspired from the social behavior of birds and animals. It is motivated by the concept of social behavior simulation and evolutionary algorithms. Its working is different from other evolutionary techniques [22]. PSO mechanism solves the optimization problems giving good results with good performance.

In PSO, every single result is a "bird" in the global search space known as "particle" or set of a software agent. Particles search for best-optimized results to assign values in a continuous form. Every particle has a solution to the given problem and selects how to move into search area with the help of own and neighboring particles experiences. Practically, initially, every particle is assigned an initial position randomly with an initial velocity. The location of particle shows a result of the problem. So, it has a value assigned by objective function [2]. Fitness value or fitness function measures the performance of particle. During search space movement operation, particles store the location of best result in their memory as it was searched. Every particle computes its best position 'pbest' (Fitness value of local) and 'gbest' (Fitness value of an entire group or global) is the best position of the whole group. At the end of every iteration, movement of every particle having some velocity is computed based on three weighted parameters like

i. The velocity drives particle towards a location in a given problem domain in which previously best optima is found.

ii. Current velocity of particles.

iii. The particle moves to an area in problem domain in which its siblings found best optima so far.

It is a useful tool for engineering and research area. It has not any mutation and overlapping operation. Search depends upon the speed of the particles. Only those particles transmit information to other particles which are optimistic during the development of generation that makes faster processing speed. PSO has simple computation as compared to other evolutionary techniques and has bigger optimization ability. PSO accepts the code in real number form and takes output directly as an optimized solution. The number of dimensions is equal to the constant value of the solution. It suffers easily from the partial optimism that creates a slow irregular speed and indirection or wrong path of particles. This algorithm is not operatable with scattering problem and is difficult to optimize [2]. Also, cannot work out with non-coordinate system problems

because of the result of the energy field and movement rules of particles in the energy field. Every step of the algorithm produces a new particle generation as described below:

1. First set initialization of particles
2. For each particle compute fitness value
3. If the fitness value is better than the best value
4. Then set current value as pbest
5. Select the particle with best fitness value as gbest
6. For each particle evaluate particle velocity
7. Update particle position and particle velocity

A complete processing of Particle Swarm Optimization is as shown in Figure 1.2.

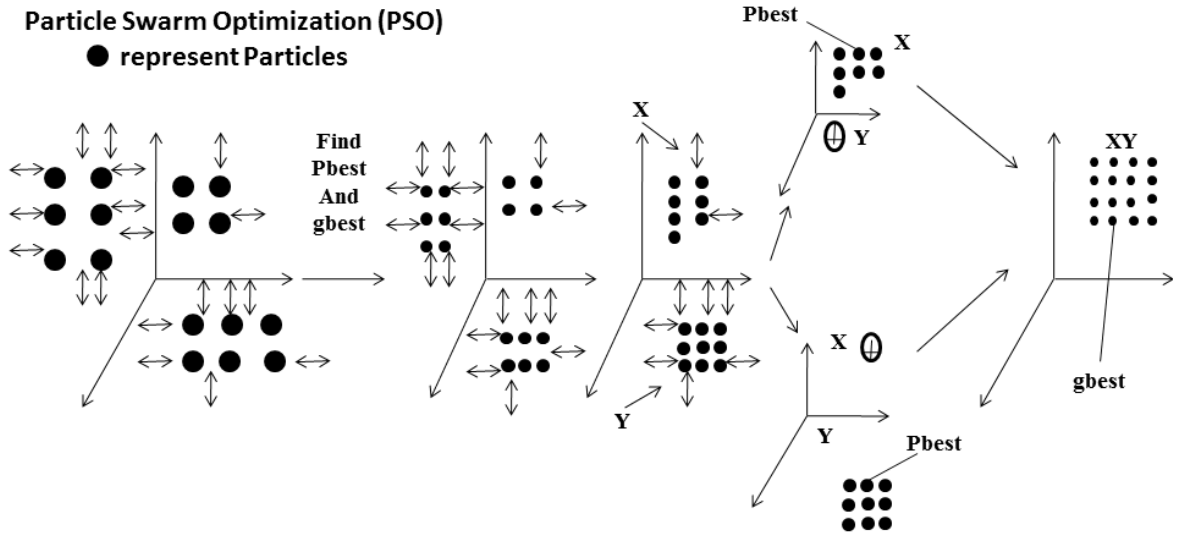

Figure 1.2: Particle Swarm Optimization process [2]

III. Variable Neighborhood Search

Variable Neighborhood Search (VNS) is a meta-heuristic mechanism to solve global optimization and combinatorial problems [3]. It has the structure of various neighborhoods having the incumbent possibilities that move from one location to another as:

i. Shaking of particles

ii. Descent to search a local optima

iii. A perturbation step to finding the optimum local area

Some fields use the concept of VNS like scheduling, cluster analysis, location theory, network design, vehicle routing, artificial intelligence, lot-sizing, pooling problem, engineering, phylogeny, biology, geometry, reliability, telecommunication design, *etc*. In this approach, heuristic first starts with some feasible options and search for a better result with moving into the neighborhood of current result. Limitation of this technique is its immature convergence.

If trapped around local optima, then moves in the neighborhood area of that local optima. Overall, it escapes from the defined area. This mechanism converges into a local optimum fast. Large size problems become a challenging issue to trap into local optima with high value. Higher the search space more is the trapping into local optima. To control this issue, the meta-heuristics approach applied and a design of heuristic approach having powerful set-up taken in the problem area. It is currently developed meta-heuristics technique in the literature to solve optimization problems and gives an incredible performance [3].

A complete process of Variable Neighborhood Search is as shown in Figure 1.3. VNS declare a neighborhood structure and are transferring particles systematically within a local search. When local search occurs within local optima with respect to the neighborhood, then it is VNS technique. So, this structure is additive to every particle during a flow of shaking and moving of particles that enhance the ability of VNS to get local optimal results. Note that using heuristic techniques, it is very crucial to design multiple neighborhood structures, but VNS solves all the issues[23]. VNS has the following major ideas:

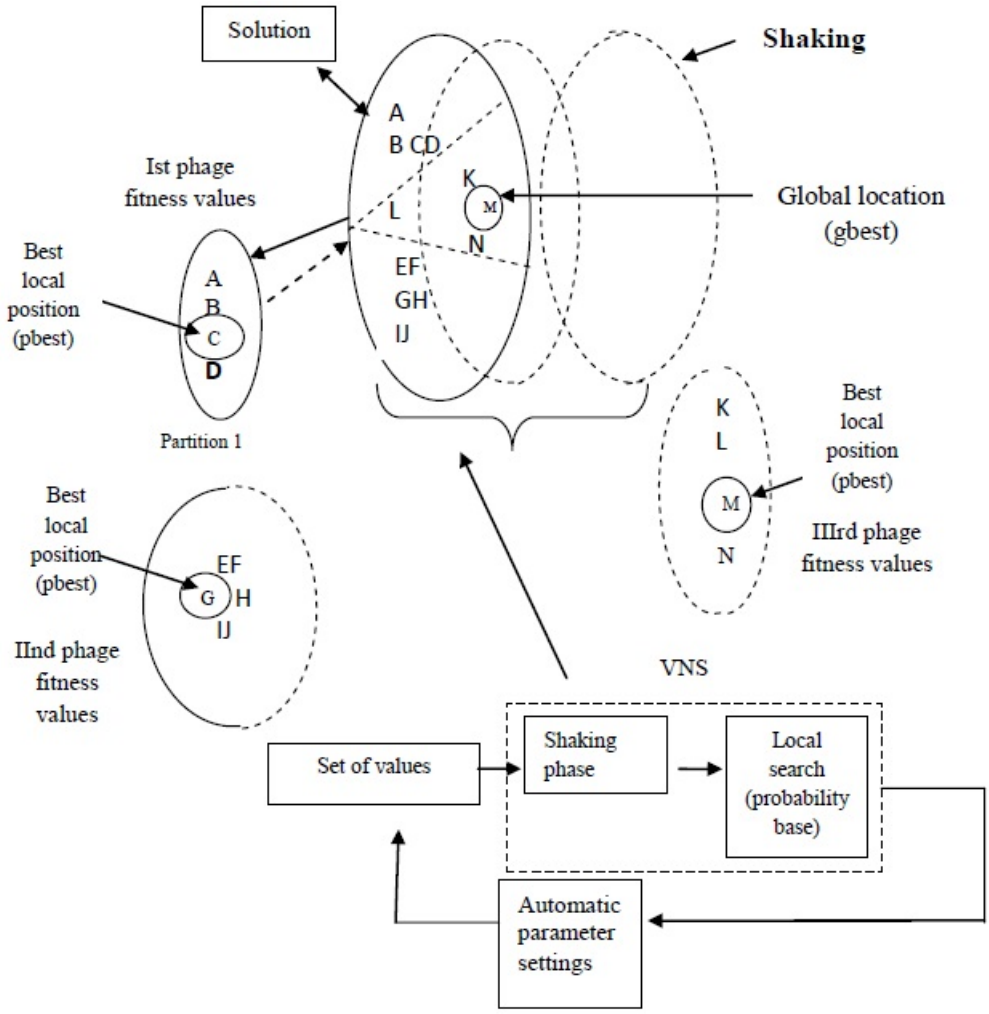

Figure 1.3: Variable Neighborhood Search process [3]

i. The local solutions in given siblings are not essentially a local solution in other siblings.

ii. A global solution is local optima for the entire set of sibling structures.

iii. In several solutions, local optimum solutions are mostly near to each other.

iv. The empirical observation states that a local solution often gives some information which belongs to a global solution.

IV. Ant Colony Optimization

In Ant Colony Optimization (ACO), some artificial ants help in building solutions for optimization problems and via a communication scheme, exchange the information. They find the shortest paths as the moving ants lay pheromone on the ground so that

when another ant encounters it, it can detect it and decide to follow the trail. As a result, the emerged collective behavior is an indication that if some ants choose a particular path, then the probability of other ants following the same path increases [24].

The steps used for ACO approach are as follows:

1. Initialize pheromone trails
2. Produce an underlying population of a solutions (ants)
3. For every ant, calculate fitness function
4. Determine the best position for each ant
5. Find the global best value (ant)
6. Update the value of pheromone
7. Process the termination step

A complete flow process of Ant Colony Optimization is as shown in Figure 1.4.

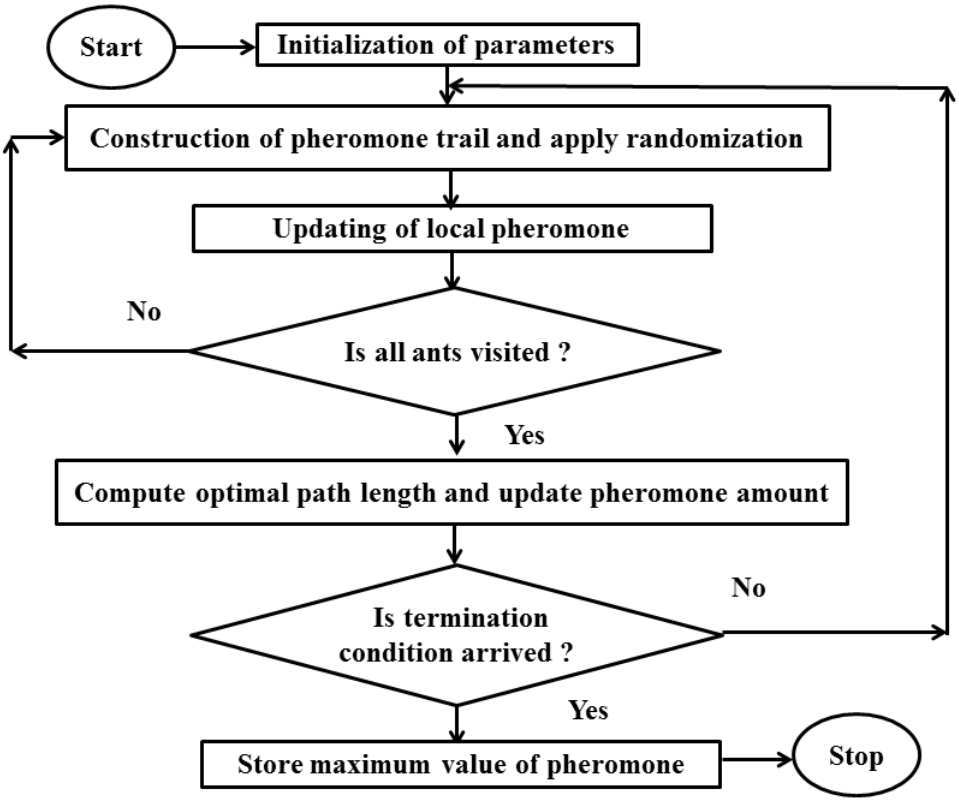

Figure 1.4: Ant Colony Optimisation Flow Process [4]

V. Artificial Bee Colony

Artificial Bee Colony (ABC) is a popular approach for optimization that simulates the intelligent foraging nature of honeybees. In this mechanism, three types of bees are considered. The first ones being the employed bees and search the food They search and share food source related information with onlooker bees. The job-shop of onlooker bees is to filter out the good food sources amongst those found by the employed bees. The highest quality (fitness) food source is more likely to be selected. The employed bees that abandon the food source and search for new ones are called scout bees [4]. A complete flow of ABC is as shown in Figure 1.5.

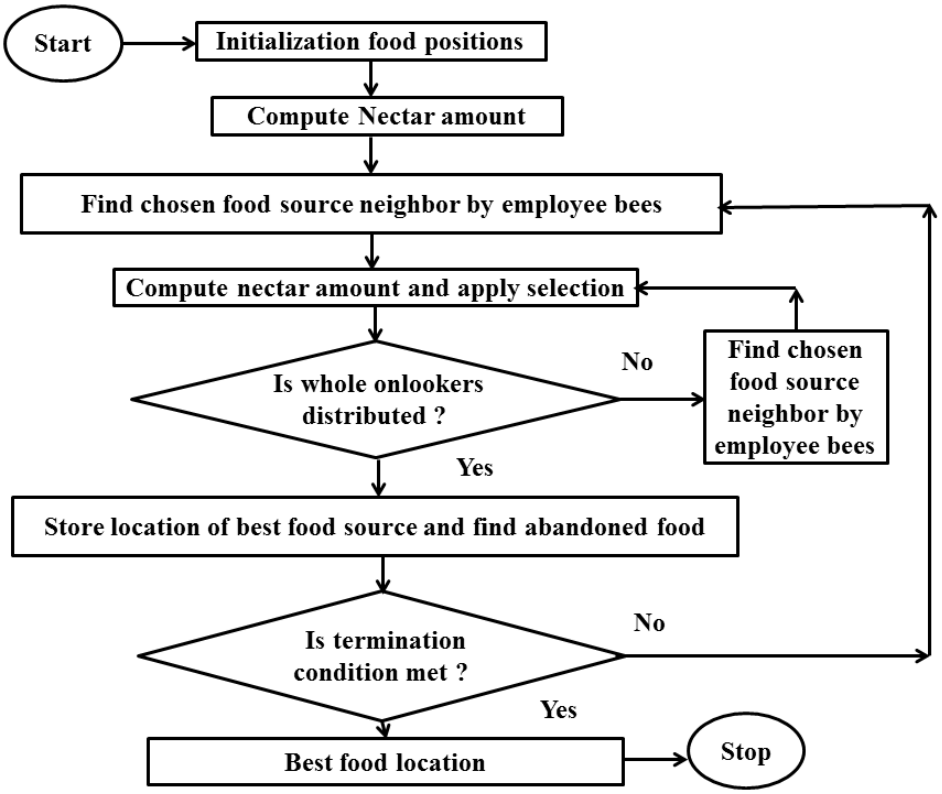

Figure 1.5: Artificial Bee Colony Flow Process [5]

The steps used to find the good food in ABC mechanism is as follows:

1. Initialize a random population
2. Evaluate its fitness function
3. Pick sites for neighborhood search
4. The bees for picked sites should be examined and the fitness function is calculated
5. From each patch, the fittest bee should be selected.

6. Remaining bees are assigned randomly to search and compute fitness value

7. Termination condition processed

It is a based on population and is a meta-heuristic technique developed from the inspiration of honeybee swarm having intelligent foraging behaviors. It has three types of foraging bees: employed bees, onlooker bees, and scout bees. Firstly, it generates a randomly distributed initial population called food source positions. Nectar amount of new one is more than that of the previous source. Bee stores the new location in memory called source position and old position removed from the memory.

Employed bees that exploit a food source currently generate updated source position in memory and verify the current nectar amount and compare it with old one. If it is higher, then store it in memory [5]. Further, the job-shop of onlooker bees to wait in the hive for information to be shared by employed bees about their food sources. The job-shop of scout bees is to search currently for a new food source in the vicinity of the hive.

Artificial Bees adopt alternative path for honey integration. They compute their current location from past trajectory path continuously. Due to bees moves back to home using the direct route as compared to backtracking a path which was the original route. So, this algorithm emerges faster and is less adaptive. It takes few steps to find and collect the food and becomes more efficient. ABC has minimum computation time for job-shop completion. Sometimes it gives poor results due to the generation of randomization sequence. It has low efficiency relatively as compared to other heuristic techniques but has powerful characteristic efficient global optimization [24].

VI. Cuckoo Search

Cuckoo Search (CS) is inspired by brood parasitism of some cuckoo species. They lay eggs in the nests of the birds of different species. Below are the three basic rules for this algorithm:
i. One egg is laid at a given time and dumped in a randomly selected nest.
ii. Only the nest having the higher quality eggs are moved to the next generation.
iii. The number of host nests is fixed and the probability of discovering the egg laid by the cuckoo by the host bird is 'P_a'. The host can discard the egg or it can abandon the nest and build a fresh one.

The steps used for CS approach are as follows:

1. Define objective function '$f(o)$'
2. Formulate the initial population of 'h' host nest
3. Eggs should be ranked after assessing the fitness
4. Get a cuckoo arbitrarily or produce new solutions by Levy flights
5. Assess fitness function 'F_i'
6. Choose a nest 'j', randomly
7. if ($F_i > F_j$) , then Replace 'j' with the new solution
8. Abandon the worst net with probability 'P_a' and a new nest is then built
9. Assess fitness, rank the solutions and find the current best

A complete flow process of Cuckoo Search optimization is shown in Figure 1.6.

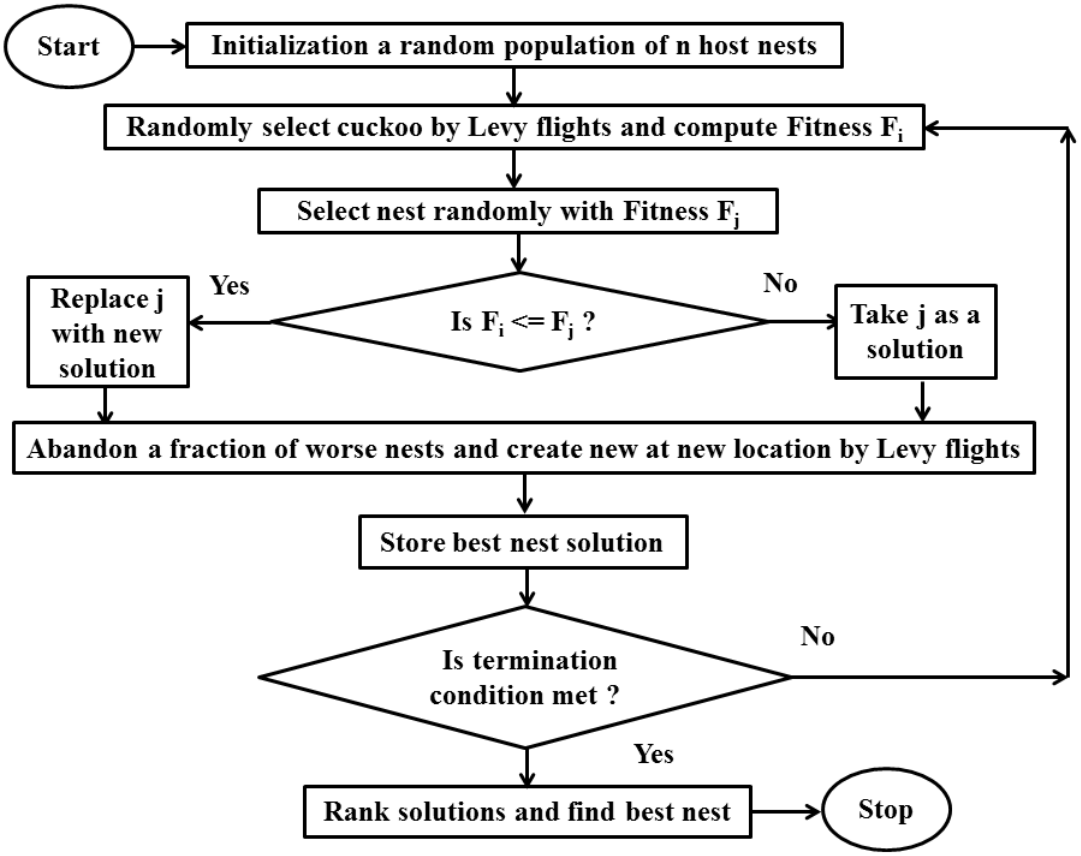

Figure 1.6: Cuckoo Search optimization flow process[6]

VII. BAT Algorithm

The standard BAT Algorithm (BA) was created by Xin-She-Yang in 2010. The main features in the BAT are based on the echo sounding nature of microbats. BAT discover their way in the night by radiating the sound signal called sonar/echolocation and use that signal to detect the object or obstacles surrounding them. They emit sonar signal very noisy so that they can listen to echo which bounces back from the obstacles in their way or the prey[7].

Most of the bats utilize echolocation to detect distance and they usually also know or guess the distinction between food or target and background obstacles in some means. Bats travel randomly with 'vi' speed at 'xi' position with a fixed frequency F_i, changing wavelength as well as loudness 'A0', to seek for prey. They are able to instantly alter the wavelength (or frequency) of their released pulse rate and modifies the speed of pulse emission 'r' in the range of [0,1], rely upon the target proximity.

Despite, the sound intensity can fluctuate in many approaches. For simplicity, the sound intensity varies from large and positive A0 to minimal steady value which denotes by A_{min}.

A complete flow process of BAT optimization is shown in Figure 1.7.

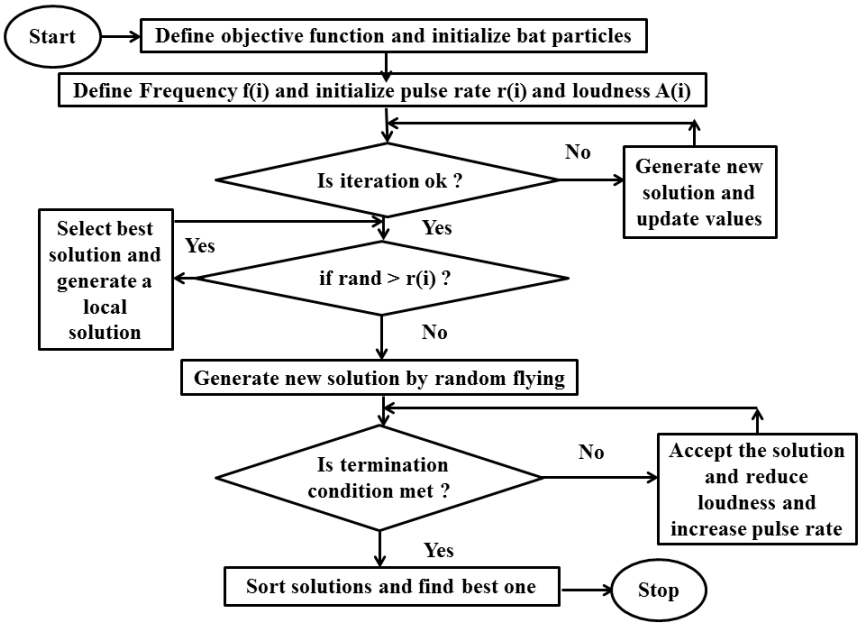

Figure 1.7: BAT optimization flow process [7]

A comparision of all the metaheuristic search techniques discussing there pros and cons is elaborated in Table 1.1.

Table 1.1: Comparative analysis of metaheuristic based scheduling techniques

S.No.	**Technique**	**Pros**	**Cons**
1	Genetic Algorithm	Finds near-optimal solution; Avoids being trapped in locally optimal solutions	No guarantee of finding global maxima; Convergence time is more.
2	Particle Swarm Optimization	Fast search speed; Calculation is simple	Tendency of premature convergence; Suffers from partial optimism
3	Variable Neighborhood Search	Descent to local optima; Use very little memory; Simple, easy to understand and implement	Stuck in local optima; Poor convergence speed
4	Ant Colony Optimization	Inherent parallelism; Can be used in the dynamic application	Time to convergence is uncertain
5	Artificial Bee Colony	High Performance; Fewer control parameters; Easily modified and hybridized with other meta-heuristic algorithms	Risk of losing relevant information; The population of the solution increases the computational cost
6	Cuckoo Search	Global convergence; Global optimality	Does not incorporate any local search to increase the convergence speed
7	BAT Algorithm	Some solutions; Frequency tuning; Simple, flexible and robust	Slow convergence speed at a later stage

1.4 Finding a good schedule

Locating a superb objective to maximize or minimize certain measures of performance is tough for the scheduling problem, it's incredible reasons. To start, important objectives the same as client satisfaction with quality or promptness are tough to quantify for proper analysis. Secondly, a store is in the main handling three different kinds of objectives, which are not the same:

i. maximize shop throughput over a long time;

ii. satisfy customer desires for quality and promptness; and

iii. minimize current out-of-pocket costs. Several ways to locate the desired objectives are possible. They are often:

iv. solve troubles with one objective before beginning;

v. solve for trade-off curves between objectives; and

vi. Combined objectives the same as minimizing the makespan and lowering the resource idle time when getting effective use resources.

A scheduling system dynamically makes a decision about matching activities and resources so that they can finish jobs and projects inside the suitable certain period of time, maximizing throughput and minimizing direct operating costs.

1.4.1 Multi-stage, Multi processor

The sort of shop scheduling problem is in general and as well most challenging to locate the desired objective. Here don't see any restrictions about the requirements linked to each task, in order that a hobby could possibly want processing of any subset of processors within the conceivable order. Mostly, the criterion is usually to reduce maximum flow time. All optimization ways to this concern are generally branch and bound procedures of Giffler and thomsans.

1.4.2 Batch Shop

A jug shop is without a doubt an empty shop with the objective the duplication in WIP (Work in Process) and final production between customers become so large that large batch processing is typical to have made use of economy of scale in processing similar parts (lot sizing). Flow through a store is not completely linear, but is usually less complex compared to closed or open shops.

1.4.3 Flow Shop

A flow shop works as a jug shop with the linear flow. The flow would be discrete, continuous or semi-continuous. Grouping and letting are typically important. In the exact easiest case, each job consists of the few activities being performed in the identical sequence on a single few machines per job. That could be, in the exact easiest case, there is only a particular machine that does the most crucial activity in each job, another that does another, and for that reason on. The most concern is a straightforward flow shop problem. In compound flow shop, each machine while using series may perhaps

be replaced by several parallel machines, which is identical or very different. Each job would flow to many of the machines in the primary cluster, then too many of the second cluster and for that reason on.

1.4.4 Assembly shop

An assembly shop is definitely an open shop or batch shop that activities join to subassembly activities, which in turn join to subassembly activities. Ultimately, this joins assembly unit, forming the final product. Labeling and cannibalization issues, which are typically crucial in open shops, remain important here also. A selected WIP item may perhaps be inventoried at numerous amount process simultaneously. Whether or not to take into account the assembly shop as distinct external to shop is ambiguous. Nevertheless leads rather naturally inside the assembly line.

Methods or techniques used for scheduling problems:

i. Branch and bound technique are one of the important technique used on large scale in optimization problems. It is commonly used for solving the combinatorial problem. This method consists of two processes of branching and bounding. In this technique, a given problem divides into a set of few new problems. The method of solving these problems is similar to the method of the original problem. The sequence problems are assigned by the points which denote the vertices and their positions denote the nodes of a tree which is called as the branching tree formed for finding the optimal solution of the problem.

ii. LPT (least production time) rule of scheduling-Here the jobs are scheduled in such a way that the time required for finishing the second job is less than the time taken by the first job that is in decreasing order of processing time.This is called LPT rule of scheduling.

iii. Two level mixed optimization method is one of the best methods of scheduling as it decides the priority of machines to be used for operations and is considered as first level.The second level decides the extra time required for particular machine for getting maximum output. This type of scheduling is used in machine shops especially run by government, as because of worker union no worker is ready to work for more than 8 hours and there is no planning of extra work schedule.

iv. Deterministic scheduling theory optimization and approximation algorithm in terms of computational complexity theory for single unrelated parallel machine scheduling, open shop, job shop, and flow shop scheduling. Machine arrangement is identical, uniform and the results of the computational theory are widely used for getting the required result. The deterministic scheduling theory links the computer theory and operational research theory.

1.5 Industrial examples

Any Manufacturing firm not engaged in mass production from the single item has scheduling problems for not less than similar compared thereto of the job-shop. Each product has some route through various work areas and machines while using factory. Together with the printing industry, time spent in typesetting the Sunday paper will depend on its length, selection of illustrations, etcetera; time spent in a printing will depend on both its length and also number printed; time allocated to binding is good bumper printed; happening time spent in packaging will be based both within the bumper printed and also book's size. Thus a printer who must schedule producing various books through his typesetting, printing, binding and packaging departments faces a four-machine flow shop problem; for a department is actually a machine and all the jobs (i.e. the books) flow from typesetting to printing, then to binding and ultimately to packaging.

Within the community of open shop scheduling, look at a large automotive garage with specialized shops. A motor vehicle may necessitate the below works: replace exhaust pipeline and muffler, align wheels and tune-up. These three tasks may perhaps be achieved in any order. However, on the reasons that exhaust system, alignment and tune-up shops are usually in many buildings, therefore it is extremely hard likely that just perform two tasks simultaneously. Here n-jobs can be taken up first and processed most of the machine centers, as there isn't any specific sequence as replacements to do the job in open shop scheduling. In these kinds of example preemption usually isn't desirable. These problems are interesting from theoretical angle also.

The run the place shuttle would perhaps be somewhere between a closed and a clear job shop. A manufacturer of a mainframe computer is often a confusing mixture of custom orders and repeat orders. A tractor house builder is a wonderful instance of a clear shop. A just to illustrate for a batch shop is perhaps an oil refinery.

1.6 Identical parallel processor scheduling

The performance measures active inside the Identical Parallel Processors are as follows; makespan, total tardiness, many tardy jobs, maximum lateness *etc*. The schematic representation of the average identical parallel processing environment involving three identical machines, required to supply eight jobs, is shown in Figure 1.2. Most of Identical Parallel Processor (IPP) scheduling processor problems are NP-Hard and that happens to be computationally challenging. Hence the building of accurate and efficient heuristics could be expected to unravel large instances of these problems. The sphere notation introduced by Lawler et al. α, β and γ was used throughout the work to spell out the cutter characteristics, job characteristics as well as the defined performance measure.

1.7 Various heuristic search algorithm

When using the growing uncertainty and complexity around the modern manufacturing system environments, use many of the scheduling complaints are became NP-Hard (that is, computational requirements grow exponentially being a function of, however, the drawback size).

Moreover, the perfect space for almost any scheduling concern is discovered to be difficult to pinpoint for devising a way to get to the suitable or maybe a near-to-optimal solution around the complex solution space. Hence, however, the drawback necessitates the sufficient and appreciable quantum of research effort to achieve out its goal. Reported by its tradition, optimization algorithms and approximation techniques have played a crucial role in solving complex scheduling problems.

The optimization algorithms include various enumerative procedures and mathematical programming techniques (such as Linear Programming, Integer Programming, Goal Programming, Transportation model, Network Analysis, Dynamic programming, Game theory, Sequencing Models, and Geometric Programming). These techniques are widely known as worthwhile for solving smaller size and over-simplified problems and located always reasonably well in providing satisfactory optimal solutions. Attributable to complexities around the varieties of scheduling problems, these techniques are not trusted upon satisfactory solutions.

On the whole, the approximation techniques add the Implicit Enumeration Technique (Branch and Bound Algorithm BBA, Decomposition method, Lagrangian Relaxation LR, Priority rules, Heuristics), Local Search algorithms (Iterative Search ITS, Simulated Annealing Algorithms SAA, Threshold Annealing TA, Variable Neighborhood Search Algorithm VNS and Tabu Search TS) and Evolutionary Algorithms (Genetic Algorithm GA,

Particle Swarm Optimization PSO, Harmony Search Algorithm HSA). Especially these techniques are discovered to be robust and yield financial success, their performances are satisfactorily supplying the objectives defined around the manufacturing systems and operating conditions remain same.

As use many of the heuristics employ efficient local search algorithm, they are known to produce excellent results, however, are susceptible to put together struck with local entrapments. The exploration relates anywhere int he planet search plus the exploitation relates to local search. The exploration is made up of probing a much bigger portion of this search space with the expectation of finding other promising solutions which are not refined. This operation amounts to diversify the search to prevent getting the trapped interior of a local optimum.

While with the exploitation, it truly is made up of probing a smallish (but promising) region of this search space with the expectation of improving promising solutions we

have definitely at hand. This operation amounts to intensify (refine) the search around the vicinity of solutions. Owing to the telltale substantial characteristics and features about heuristics, they can indeed be more spent on the investigation for solving complex scheduling problems. This particular research work, few the completely search heuristics are fine-tuned and tailor-made to devise intelligent mechanisms to build satisfactory schedules for IPP scheduling problems.

These approximate goals are necessary because it is hard to define long-run profit in the short-run situation where it often appears that all costs are fixed. Results of the scheduling activities are fed back to the other planning and control areas to improve their decision making.

To differentiate between schedules and to select the best one, we have to have some measures of effectiveness, as in other areas where we want to “optimize”, with which we can compare the different solutions.

In general, we want to minimize either the length of operation time such as total processing 8utime, the completion time for certain products, average finishing time, total project time, minimize idle time, or we want to minimize certain costs such as the unit cost of production, total cost, *etc*. Examining the different measure of effectiveness more closely we find that some of them are not applicable for certain problems and that, what is worse, many of them are contradictory. Here we face "the dilemma of scheduling", which is particularly evident in job shop production. The following points are examples of the contradiction in the scheduling operation:

i. We want to decrease the average in-process time of our work orders, thus decreasing in-process inventory and increasing the likelihood of meeting due dates.

ii. Also, we want to increase the degree of utilization of equipment, thus increasing the return on our investment in physical facilities.

Achievement of the first goal would lead to the selection of the schedule with the smallest in-process time for all our products, such as the processing time, the transportation time, the waiting time, and the setup time. This objective focuses on the jobs to be done and implies moving them rapidly through the production process. To achieve the second goal, we need to pick the schedule that maximizes the utilization of existing capacity. This objective focuses on the machines and implies arrangement of jobs to suit the machines.

1.8 Thesis organization

After the introduction to the thesis in Chapter 1, the rest of the thesis is structured as follows:

Chapter 2 discusses the literature survey in the area of machines computing, job shop scheduling. It also lays out a discussion on the job shop duplication based scheduling techniques. Finally, it outlines various meta-heuristic techniques and demonstrates the limitations of existing scheduling techniques.

Chapter 3 discusses the mathematical formulation of job-shop scheduling. Then, gaps in the literature are formulated. Afterward, problem definition and objectives are defined.

Chapter 4 discusses the proposed job-shop scheduling technique.

Chapter 5 demonstrates the experimental set-up for simulating the proposed techniques. It also describes the required number of attributes along with their tuned values. Experimental results have been described in the tabular and graphical forms.

Chapter 6 gives the concluding remarks on the research by highlighting the significant contributions of the work done. Future directions of the implemented research work have also been presented towards the end of this chapter. The chapter concludes the thesis, by explaining briefly, the outcome of each of the chapter.

Chapter 2

Review of literature

Outline

This chapter reviews job shop scheduling techniques and its basics, followed by a detailed survey of existing meta-heuristic techniques. Finally, it outlines comparison between various meta-heuristics based scheduling techniques.

The continuous changing demands of the purchasers and the desire to deliver better quality products are considered for the driving factors to improve the performance of manufacturing systems. The manufacturing system may need to enjoy the adaptability to achieve the necessary requirements of this manufacturing systems as soon as possible. Within the last three decades, these manufacturing systems have attained an amazing growth by introducing and implementing various managerial concepts and various tools that control the fabrication procedure based on backyard garden production environments and also coordinating the desired communications between various levels of organization using various advanced information technology. Various methodologies for solving identical parallel machine scheduling troubles are discussed in section 2.2. In section 2.3 scheduling depending on machine breakdown is discussed. Multi-objective scheduling is discussed in section 2.4. Research gaps and conclusions are discussed in section 2.5.

2.1 Identical parallel processor scheduling

The identical parallel machine scheduling troubles are solved using classical mathematical methodologies like dynamic programming methods, mixed integer linear programming *etc*. While thinking about the meta-heuristics, various meta-heuristics have been completely applied to fix identical parallel machine scheduling problems. The

meta-heuristics, Tabu Search (TS), Simulated Annealing Algorithm (SAA), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Harmony Search Algorithm (HSA) are viewed as since several attracted heuristics among the many researchers. The taxonomic framework for scheduling is depicted in Figure 2.1.

2.1.1 Tabu Search

Kim et al. (2003) [25] presented a restricted tabu search algorithm to schedules the jobs on parallel machines. The main objective of this technique was minimized the maximum lateness of the jobs. It was also reduced the search effort significantly while obtaining good quality final schedule.

Schaller (2014) [26] proposed an improved Tabu search algorithm for scheduling of jobs on identical parallel machines. The main objective was to reduce the overall tardiness. In this technique, genetic algorithm was also used to obtain better results.

Li. et al. (2016) [27] designed an algorithm in order to reduce the makespan. GA and TS were used to schedule the jobs inflexible manner.

Kuhpfahl et al. (2017) [28] suggested the use of heuristic in scheduling of jobs to reduce the overall tardiness. In this technique, local search algorithm was utilized to enhance the ability of heuristic techniques to make scheduling more efficiently.

Nouiri et al. (2018) [3] utilized PSO algorithm to carried out flexible job shop scheduling. The main objective of this algorithm was to maximize the throughput.

Rameshkumar et al. (2018) [2] studied the use of PSO algorithm in job shop scheduling. The main objective was to reduce the makespan time.

2.1.2 Simulated Annealing Algorithm (SAA)

Melouk et al. (2004) [29] suggested SAA that reduce makespan time for scheduling the non-identical jobs on single machine.

Lee et al. (2006) [30] addressed the problem of maximum execution time. It used SAA to schedule the jobs on machines in parallel fashion in order to reduce the makespan and maximize the execution time.

Damodaran (2012) [31] utilized SAA algorithm to reduce the overall makespan time for parallel batch processing machines with unequal job ready times. The results showed that it provides better results than existing techniques.

Yeh et al. (2014) [32] proposed an algorithm using fuzzy and machine learning techniques to schedule the jobs on parallel machines. The aim of this technique was to minimize the makespan.

Shivasankaran et al. (2015) [33] studied immune SAA algorithm to efficiently schedule the jobs. This technique maximized the speedup and utilization of machines.

Kuhpfahl et al. (2016) [28] improved the utilization of machines by using local search neighborhood algorithm for job scheduling. It effectively reduced the overall tardiness of machines and enhance the efficiency.

Dao et al. (2018) [34] proposed an algorithm using BAT to optimize the makespan of parallel machines for job-shop scheduling. It was also minimized the energy consumption and waiting time.

2.1.3 Genetic Algorithm (GA)

Rajakumar et al. (2007) [35] utilized GA to balance the load on parallel machines for job scheduling. It also enhanced the efficiency of machines.

Tseng et al. (2009) [36] studied GA and local search algorithm to solve the job-shop scheduling problem. The permutation operation was used to assign the jobs to machines. It minimized the makespan time of machines.

Chaudhry et al. (2009) [37] given a novel method using GA which eliminates 100 % tardiness including few jobs on identical machines. It was able to balance the load efficiently among available machines.

Tavakkoli et al. (2009) [38] designed an algorithm using GA for unrelated parallel machines scheduling with bi-objective fitness. The main focus of this algorithm was to reduce the tardiness of machines to enhance the efficiency.

Balin et al. (2011) [11] dealt with job-shop scheduling problems by means of GA. To enhance the speed, fuzzy technique was also utilized. It reduced the makespan time and load balance rate.

Chen et al. (2012) [8] created a management protocol motivated by simply Inherited Algorithmic rule regarding job browse management concern by means of analog appliances and reentrant process.

Zhang et al. (2013) [39] suggested a new rescheduling technique using GA and TS which dynamically manage the jobs when breakdown of machines occur.

Spanos et al. (2014) [40] proposed parallel GA to manage the jobs on identical machines. The concept behind to use parallel GA was that it enhances the speed up time. The classical GA is time-consuming technique which adversely affects the overall performance of scheduling algorithm.

Zhang et al. (2016) [41] solved the energy consumption issue related to job shop scheduling. To solve this issue, GA with enhanced local search was used using multi-objective fitness function. It minimized the total weighted tardiness and total energy consumption.

Peng et al. (2018) [42] used GA for double-resource flexible job shop scheduling. Experimental results showed that the proposed technique provides better utilization and throughput as compared to other techniques.

2.1.4 Particle Swarm Optimization (PSO) Algorithm

Torabi et al. (2013) [43] designed multi-objective particle swarm optimization (MOPSO) for parallel machines to sole job-shop scheduling problem. It minimizes the load balance rate and overall tardiness.

yin et al. (2014) [44] solved the job-shop scheduling algorithm using discrete PSO. TO reduce the tardiness, local neighborhood search was applied. The results showed that it better performance as compared to existing techniques.

Muthiahet al. (2016) [5] used the combination ABC and PSO to reduce the makespan around the shops. The optimal makespan fitness feature exactness around the consist of strategy was 94.23 % in comparison as nicely optimization processes.

Wu CC, et al. (2018) [45] designed a two-stage three-machine assembly flow shop scheduling with learning consideration to minimize the flow-time by six hybrids of particle swarm optimization. But, it suffers from the load balancing issue.

2.2 Multi-objective scheduling

Recently, many multi-objective optimization based techniques have been designed to achieve an efficient scheduling results. In it, generally Pareto optimal solutions are used to achieve the desired goal.

Ruiz et al. (2007) [46] used recommended based multi-objective job-shop scheduling technique to reduce the energy consumption and makespan time in an efficient way. It has efficiently balanced the workload between available machines.

Liang et al. (2013) [47] designed an important variable local community query (VNS) algorithm along with an array of pismire settlement SEO algorithm to resolve an identical parallel appliance preparation challenge with several inconsistent plans: generate period together with total tardiness.

Zarandi et al. (2015) [48] utilized Non dominated sorting based genetic algorithm (NSGA)-II to schedule the jobs between available machines.

Zhang et al.(2016) [41] implemented a novel multi-objective optimization based job-shop scheduling technique. The main objective of this technique was to minimize the makespan and energy consumption and to maximize the efficiency of machines.

Zhao et al. (2018) [49] designed a two-generation Pareto ant colony algorithm for multi-objective job shop scheduling problem with alternative process plans and unrelated parallel machines. However, it suffers from poor convergence speed issue.

Chapter 3

Problem formulation

Outline

This Chapter defines the mathematical formulations of the proposed techniques. The significance of proposed technique is also discussed. Various performance metrics which have been used in this thesis to compare the job-shop scheduling techniques have also been described.

3.1 Gaps in literature

This section describes various gaps that are identified in the literature. Majority of existing researchers have neglected at least one of the following issues:

1. **Stuck in local optima:** Stuck in local optima is found to be a major drawback of majority of existing meta-heuristic based job-shop scheduling techniques. It can be successfully solved by considering the hybridization of meta-heuristic techniques with other meta-heuristic techniques.

2. **Convergence speed:** Majority of existing metaheuristic based job-shop scheduling techniques suffer from poor convergence speed issue such as GA, ACO, VNS *etc*.

3. **The initial amount of particles:** From the literature review it has been found that PSO variants suffer from the initial number of particles selection issue. It states that poorly selected particles may lead poor results most of the time.

3.2 Problem definition

Scheduling is concerned with the problems of optimally allocating available resources to process a given set of jobs. In particular, scheduling jobs on multiple machines has received extensive study in the past four decades, in computer science, operations research, and system sciences

As per the literature review, one of the main issues in parallel environment is efficient scheduling of jobs. Job-shop scheduling is required to schedule the workload between machines. An efficient job-shop scheduling technique has an ability to reduce the makespan and energy consumption of machines. Job-shop scheduling helps to achieve high user satisfaction and resource utilization ratio. With proper job-shop scheduling, hot-spots can be prevented and resource utility levels can be improved.

Therefore, job-shop scheduling techniques can further reduce energy consumption of machines thereby, reducing carbon emission and cooling requirements of the parallel machines, which is a dire need of machines computing. The review of existing job-shop scheduling techniques has shown that the majority of existing job-shop scheduling problem is NP-hard in nature. Therefore, there is a need to develop a near to optimal job-shop scheduling technique that can improve the performance of parallel machines by assigning the workload between machines in an efficient manner.

Recently, several researchers have solved job-shop scheduling issue by considering well-known meta-heuristic techniques. These techniques are Genetic algorithm (GA), Particle swarm optimization (PSO), Variable neighborhood search (VNS), Ant colony optimization (ACO), BAT algorithm, Artificial bee colony *etc*. However, these meta-heuristic techniques suffer from one of these issues: premature convergence, poor convergence speed, initially selected random particles and stuck in local optima issues. Additionally, majority of existing researchers have considered scheduling of independent jobs only.

To handle these issues a hybrid meta-heuristic based job-shop scheduling techniques is proposed in this work. Hybridization of ACO and GA based technique has overcome numerous issues associated with existing techniques. The comparisons will also be been drawn between proposed and existing techniques by considering some well-known parallel performance metrics.

3.3 Objectives

To remove the issues associated with existing meta-heuristic based job-shop scheduling techniques, following objectives have been formulated:

1. To study and analyze the performance of ACO based job-shop scheduling technique.
2. To design and implement a hybrid meta-heuristic based job-shop scheduling techniques using ACO and GA.
3. To compare the proposed techniques with existing meta-heuristic techniques by using the following performance metrics:
   - Makespan time in seconds
   - Average execution time in seconds
   - Energy consumption in joules

3.4 Performance metrics

This section describes various well-known quality metrics which are used to evaluate the performance of job-shop scheduling techniques. These metrics are as:

3.4.1 Makespan

Makespan is a well-known quality metric used in parallel and distributed computing. It determines the time till the entire jobs running on all machines get executed. It is computed by taking the maximum value of schedule length of each VM$_i$. Therefore, makespan should be minimized. Makespan (ms) is computed as follows:

$$ms = \max\left(sl\right) \tag{3.1}$$

Here, sl is the schedule length of each VM$_i$.

3.4.2 Energy consumption

Energy consumption is another performance metric used to evaluate the energy consumed by parallel machines. It is measured in Joules (N_j). It should be minimized. Energy consumption is computed as follows:

$$EC = \lambda \times I_t + \delta * B_t \tag{3.2}$$

Here, I_t and B_t represent idle and busy time of VM$_i$, respectively. Also, λ and δ denote energy consumption for idle and busy time of VM$_i$, respectively.

3.4.3 Execution time

Average execution time (AET) is the proportion of time spend during the computation of jobs and communication between machines. AET is computed as:

$$AET = \frac{\sum_{i=1}^{N} E_x^i}{N} \tag{3.3}$$

Here, E_x represents execution time of each machine. N defines total number of machines.

Chapter 4

Proposed technique

Outline

In this Chapter, machines model has been designed along with its fitness function. The primary objective of this chapter is to design and model new job-shop scheduling techniques to minimize designed objective function for machines data centers. For efficient execution of user jobs, different scheduling techniques have been proposed in this Chapter. These are (a) Genetic variable neighborhood search with job-shop duplication, (b) Multi-objective genetic variable neighborhood search using job-shop duplication, (c) Hybrid meta-heuristic based scheduling with job-shop duplication and (d) Hybrid genetic-variable neighborhood search with particle swarm optimization.

4.1 Job shop scheduling problem

Job shop scheduling is an optimization problem and in JSSP resources are allocated to perform a collection of jobs at a particular times. In JSSP Resources are allocated to perform a collection of jobs at particular times. In JSSP, there are m number of jobs i.e $J_1, J_2, J3....J_m$ each of the jobs having different processing times. Which is to be processed on m number of machines each having different processing speed considering every single job-shop must schedule once on a machine. The main aim of job-shop scheduling problem is to reduce the makespan with the best machine processing sequence for all jobs. In 1966,Graham presented competitive analysis for the combinatorial optimization JSSP problem. In this work, Talliard benchmarks gives the best problem instances for make span objective for basic models [10].

4.1.1 Description of the Problem

1. There is m number of jobs which is to be processed on n number of machines.
2. Every job-shop$i$ takes J_i precedence-Constrained operation to be performed.
3. On the number of substitutes (alternate non-identical) operation j of task $i(O_(ij)$ is processed and $t_(ijk)$ is its processing time which differs from the features of the machine k.

This gives the multiple routing of jobs. If a machine tool is overloaded for a limited period of time while the other one is idle.where capacity problem arises, substitute routing is useful there. Assumptions To solve Job Shop Scheduling Problem the following are the assumptions and constrained to be considered are

1. All the jobs are different
2. Machining time includes job-shop setup time.
3. Job shop description is already known
4. All the machines are frequently available.
5. There is no break in between job-shop processing.
6. Machines are not able to process parallel jobs at a time.
7. Each of the machines must process a job.
8. To complete the process, each of the job-shop requires m machines.
9. Preemptions are not allowed. The processing order is not same and
10. Operations are not interrupted.

There are many optimization techniques which have been applied to solve JSSP. Some of the traditional methods such as Lagrangian relaxation and branch and bound are based on traditional methods like numerical search and mathematical model can persuade the optimum solution. To solve JSSP, these are the methods which have been used efficiently and effectively. These methods are used to solve large computational problems like problem size of 10 $\times$ 10 and Problems like JSSP but there is a computational limit exist because it consists of high computational time resources.
[50] Earlier using approximation optimization method or meta-heuristic method(like Simulated annealing and tabu search, in there iterative or search process these methods follow stochastic steps.), Larger size problems like JSSP has been solved. However

optimum solutions are not guaranteed by these methods.

There is a 3×3 job-shop scheduling problem which is to be performed on 3 jobs and 3 machines as illustrated in Table 4.1.1.

In this example, there is a pre-defined machine sequence on which that three operations are to be processed. On machine N_1,the first job-shop(J_1) is to be operated initially for time units 10 and then on machine N_2 and machine N_3 job-shopJ_1 is to be processed sequentially for time units 9 and 8 respectively.The job-shopj_2 have to be initially performed on N_3 for 9 units of time and sequentially for the after jobs N_1 and N_2 for 8 and 7 units of time, respectively. similarly for J_3 the third job-shop to be performed on three machines with scheduled processing time. In order to optimize single or multiple scheduling objectives, our main task is to schedule all the jobs to search best schedule(s) for utilizing all pre-defined jobs.which is used to get the best of schedules alike the minimization of the makespan (C_{max}).

Job	Operation	Time	Machine(N_k)		
	O_{ij}	t_{ij}	N_1	N_2	N_3
J1	O_{11}	10	10	-	-
	O_{12}	9	-	9	-
	O_{13}	8	-	-	8
J2	O_{23}	9	-	-	9
	O_{21}	8	8	-	-
	O_{22}	7	-	7	-
J3	O_{33}	10	-	-	10
	O_{31}	8	8	-	-
	O_{32}	11	-	11	-

Table 4.1: Three-jobs-three-machines scheduling problem example with processing time

4.2 Ant colony optimization

ACO algorithms are inspired by the hunting(foraging) behavior of natural ant colonies. Pheromone is a substance which ants deposits individually while moving from nest to food sources and vice versa. So, the other ants go after the deposited pheromones to find the path. Therefore the vigorous pheromone in a path has the highest probability of choosing that path.accordingly, all the ants will follow the shortest path from nest to food. Every single ant works as a computational agent where an ant colony based technique uses these agents to guide the search mechanism for optimality in the solution

space [51].

Considering previous concept, Ant chooses one of the nodes and each artificial ant starts with an empty position. Until a complete tour is built, an ant iteratively fixes an unsolicited node to the partial tour constructed. At each particular step, a node is selected by using a transition rule depends on the pheromone trail to guide(lead) the other ants as local pheromone updating rule. Use the performance quality of constructed tour. When all ants in the colony have created their tours, then the search gets more directed by applying a global updating rule, the pheromone trails are modified. The methodology of the proposed ACO based algorithm is described as follows.

4.3 Methodology

In Scheduling, The Job Shop Scheduling problem made up of m number of jobs having varying processing times with n number of machines.where all the machines are same and flow of working of each job shop is unidirectional. While examining job shop scheduling problem it is supposed that the order in which all machines process each job shop is same on all machines. This kind of schedule is known as permutation schedule and is described as complete sequence of all jobs. Following terminologies are used in the proposed methodology:

m	number of jobs which is to be schedule.
n	number of machines in job shop scheduling
P_{ab}	activity time of job-shop a on machine b
$t_{a'b}$	build up time on machine b,when job-shop a'
α	Required set of job shop already processed out of n number of jobs
l	last job shop in α
$q(\alpha, b)$	finish time of partial sequence α on machine b
$\alpha(a, b)$	finish time of job-shop a on machine b,when partial sequence α has subjoin the job

By this recursive equation,the finish time of αa on machine b can be found.

$$q(\alpha a, b) = max(q(\alpha, b) + t^{b}_{l,a'}; q(\alpha a, b-1)) + P_{ab} \tag{4.1}$$

where $q(\Phi, b) = 0$and$q(\alpha, 0) = 0$ for all b and α,a null schedule is denoted with Φ for all machines,it is supposed that $t_{\Phi a}$ exits for all the jobs. It is also supposed that when each of the job shop is not obtainable at the machine then also can do setup of a

machine.

The sequence time of job-shop$a$, C_a is given as follows:

$$C_a = q(\alpha_a, n) \tag{4.2}$$

Makespan M is achieved as follows when each task is scheduled.

$$M = maxC_a, a = 1, 2, 3....., m. \tag{4.3}$$

4.4 Ant colony based job shop scheduling

This section provides step by step detail of the ant colony based job-shop scheduling technique.

Step 1: Start the pheromone trail and specifications

Step 2: closing condition is not meeting. While(if loop is not closing) apply the steps given below

- create a solution
- item using Local Search upgrade the solution
- Amend the perfume concentration,Symbolize as τ_{ap}, where $\tau_{max} \geq \tau_{ap} \geq \tau_{min}$

Step 3: return best solution found.

In the reference of the relevance of ACO algorithm to Job shop scheduling problems, τ_{ap} stands for trail intensity of framing job-shop a in position p of an ordering.for each iteration of every job-shop a of position p, pheromone value is updated and stored. In the accordance to assure that during the process of restoring(update) the trail, the trail intensities would not go above bottom line certain limits.For τ_{ap},The Maximum and minimum value indicated by τ_{max} and τ_{min},respectively as the MMAS introduced.

4.4.1 Parameters initialization

At first,all τ_{ap} are set equal to τ_{max},where τ_{max} is equal to $\tau_{max}/5$. Oberseve that persistence of the trail is denoted by ρ and is set to 0.75. e$\tau_{max} \geq \tau_{ip} \geq \tau_{min}$ is kept in the MMAS search process.The best value of objective function is refers to the team M_{best} that has been obtained so far and gave a solution to the given flow shop scheduling problem with the aim of minimizing the makespan.

Modified first move strategy improved the solution by local search. In the Paper, local search procedure is referred as the position based local search procedure. M_{best} is set for the makespan solution produced by this local search procedure and τ_{ap} are initialized.

4.4.2 Ant sequence generation

By choosing a job shop for the first position, an ant starts constructing a sequence, second position for the nonscheduled job and so on. initially a dummy job shop O is established and the partial construction sequence begins. It leads to the building of complete sequence. The following procedure in case of ant colony optimization is used to select an unscheduled job, say job-shop a, for probabilistically position p.
In the range $[0, 1]$,sample a uniform random number u.
The job-shop that are not yet scheduled, choose the job-shop with top mpst value of τ_{ap} if $u \leq (m-4)/m$, else.select job shop a from the unscheduled job shop first five-set present in the best sequence acquired so far. Sampling from the following probability distribution for position p.

$$\epsilon_{ap} = (\tau_{ap}/\Sigma a^{"} \tau_{ap}); \tag{4.4}$$

Here,job-shop a" belongs to the first five scheduled jobs, as present in the best job shop sequence so far. Note that, all the unscheduled jobs are considered when there are less than five jobs which are not scheduled yet.
A complete job shop sequence is made up by making use of the above terminologies for providing a job shop to the available partial sequence, starting from a null sequence. This type of generated sequence is called an ant sequence.

4.4.3 Updation of trail intensities

To enhance the quality of solution an ant sequence is undergoing local search procedure. The makespan(the objective function) of this upgraded sequence is denoted with $M_{current}$.Later, updated trail intensities are as follows:if job shop a is placed in position p in the produced sequence;$= \rho \times \tau_{ap}^{old}$,otherwise; where,$\rho$ is set equal to 0.75 if the generated sequence is surpassing(superior) to the best sequence that has been acquired so far.

4.5 Proposed ant colony algorithm

The important feature of the PACA is proposed followed by the detailed discussion of the algorithm.

1. Initial solution obtain,Σ^*
2. by the following local search procedure, improve the initial solution.
    - limited sequence based local search.
    - general seed order based search.
    - local search on random job shop insertion. The seed sequence is taken as the final sequence for the PACA. This sequence is called Σ^*.
3. Initialize the parameters and pheromone trail.
4. while(closing condition does not meet) as follows:
    - Using principle of the PACA, generate an ant sequence.
    - If the ant sequence which is generated is same as a previous ant sequence.
    - Using random job shop fitting local search improves the ant solution with the best solution so far and upgrade the best solution, if necessary.
    - upgrade the trail intensities or pheromone trail denoted as τ_{ap}.
5. By using these following local search procedures, upgrade the best solution obtained so far.
    - limited sequence based local search.
    - general seed order based local search.
    - local search on random job shop insertion.
6. Submit the solution found.we have assumed
   Parent 1= Best solution obtained from ACO Parent2 = second best solution obtained from ACO.
7. apply genetic algorithm as a post process.
8. applies mutation and crossover.
9. apply selection operator.
10. submits the best solution found.

4.5.1 Exploration using genetic algorithm

To explore the results obtained from ACO, following GA steps are followed:

A step by step methodology is used. Initially, the machines based model is designed by considering the well-known benchmark job-shop problems. Initially, the random initialization of given set of solutions will be done by considering the concept of job-shop duplication. Afterward, the GA will come into action to optimize the random solution by utilizing the mutation and crossover operators. The subsequent section contains the detail of each technique with suitable procedures and required formulas.

To converge and for finding the better-optimized values, the GA utilizes GA operators. Initially, a set of random solutions so-called population is developed. Then, selection operator comes into action by evaluating the objective function for each random solution. In this work ms is used as an objective function. Then, selected random solutions with minimum ms are further improved by applying the crossover and mutation operators. These operators are repeated iteratively, till termination condition not met. Termination condition depends upon the number of objective function evaluations (N_{oe}). Once $N_{oe} == Max_{oe}$ met, GA will automatically return a solution.

Algorithm 1 shows various steps which are required to successfully implement HJD. Here, N_{oe} is the number of objective function evaluation.

Algorithm 1 : Hybrid job-shop duplication algorithm

1: Initialize the GA.
2: Obtain initial population from ant colony optimization.
3: Now call **Algorithm 2**, for sub-jobs assignment to high-end machines and compute respective fitness function i.e. ms.
4: Repeat step 5 and 6 $while(N_{oe} \leq Max_{oe})$
5: New set of optimized solution be obtained from given set of individuals.
6: Call the **Algorithm 3** to apply selection and crossover operator.
7: Finally call the **Algorithm 4** to apply mutation operator on selected solutions.
8: Return the final optimized schedule.

A. Generating population in hybrid job-shop duplication

Set of chromosome was created with the help of ACO as explain in earlier Section and a solution were obtained from every chromosome in population set in chromosome creation operation. Chromosomes list constructed with the help of integer numbers randomly. The random integer numbers are created by permutation technique. The encoding procedure of chromosomes having high priority job-shop list to low priority job-shop list. Here, n genes or jobs or sub-jobs (can be taken as sub-jobs also) executed or scheduled in a priority list (high order priority to low order priority) or in a sequence. Topological order is the best for arrangement of sub-jobs or jobs. So, job-shop or subjobs

are evaluated as these occur having cost matrix with initial population size $P_{size} = 4$.

B. Assignment of sub-jobs to machines

In the case of originated population, every individual should have a major priority mechanism having permutation process. Therefore, sub-jobs should follow precedence conditions for this process. A sub-job-shop will be allocated to the machine with maximum speed, if and only if it is not already scheduled. Further, it allocates given sub-jobs to the machines(s) in such a way that it minimizes the overall ms.

The initial start time (I_{ST}) of the sub-job-shopt_a on machine p_i is symbolized as I_{ST} (t_a, pi) which is obtained as follows:

$$I_{ST}(t_e, p_i) = 0 \tag{4.5}$$

$$I_{ST}(t_a, p_i) = max_{t_s \epsilon pred(t_s)} + A_{ST}(t_s) + (C(t_a, t_s)) \tag{4.6}$$

Here, t_e is job-shop entry. The actual start of sub job-shopt_a on machine p_i is symbolized as A_{ST} (Actual start time) (t_a, p_i). This is computed as follows:

$$A_{ST}(t_a, p_i) = max(I_{ST}((t_a, p_i), avail(p_i)) \tag{4.7}$$

Here, avail(p_i) is time that the machine p_i has idle and ready for the job-shop execution. The earliest finish time of sub-job-shopt_a on machine p_i is symbolized as $I_{FT}(t_a, p_i)$ which is obtained as follows.

$$I_{FT}(t_a, p_i) = CC(t_a, p_i) + A_{ST}(t_a, p_i) \tag{4.8}$$

Here, $CC(t_a, p_i)$ is the computational cost of the sub-job-shopt_a on machine p_i . The actual finish time A_{FT} is computed as follows.

$$A_{FT}(t_a, p_i) = \min_{1 \leq l \leq P} E_{FT}(t_a, p_i) \tag{4.9}$$

Allocation of jobs or sub-jobs to n machines or using the job-shop duplication criteria is a significant achievement in proposed technique and procedure is described in Algorithm 2.

Here, JDR is the job-shop duplication rate having range 0.04 to 0.07 for experiment execution and note the final value of job-shop duplication rate on achieving the better optimized makespan. By doing this procedure there is 50% to 60% chance for duplication of jobs on the available machines.

Algorithm 2 : Job-shop allocation

Initialize current population values
Input the chromosome size value as P_{size}
Evaluate schedule length or makespan ms from a priority queue list of jobs or subjobs allocated PQL
While $(PQL \neq Null)$ **do**
    First job-shop or subjob-shopfrom PQL is selected
    For machines $pi = 1$ to n **do**
        Evaluate Ft or (ms) as fitness function using HEFT scheduling process
        If $rand \leq JDR$ **then**
            Assign i^{th} subjob-shop to all virtual machines
            Evaluate maximum schedule length or makespan (ms)
        Else
            Allocate i^{th} subjob-shop to j^{th} machines
            evaluate maximum schedule length or makespan (ms)
        End If
        Assign jobs or subjobs to virtual machines
        Evaluate $ms = \max$(schedule length)
    End For
    Remove i^{th} subjob-shop or job-shop from PQL
End While
Return the final value of makespan ms

C. Crossover operation

It is very difficult for offspring creation operation that has the same characteristics as their parent. For better computation and getting a change in the chromosomes list, crossover operator is best utilized. To strengthen the heritability of the crossover operation. According to current research [21], the role of crossover operator in this original representation was transformed to the permutation in this evolutionary technique that made the change in population. Crossover rate helped the parents side and child side diversity for optimal solution. This created four siblings from two sets of parents having diversity in the solution set. So, optimized solutions obtained in the crossover process is shown in Figure 4.1. The evaluation and replacement of offsprings with parents are illustrated in Algorithm 3.

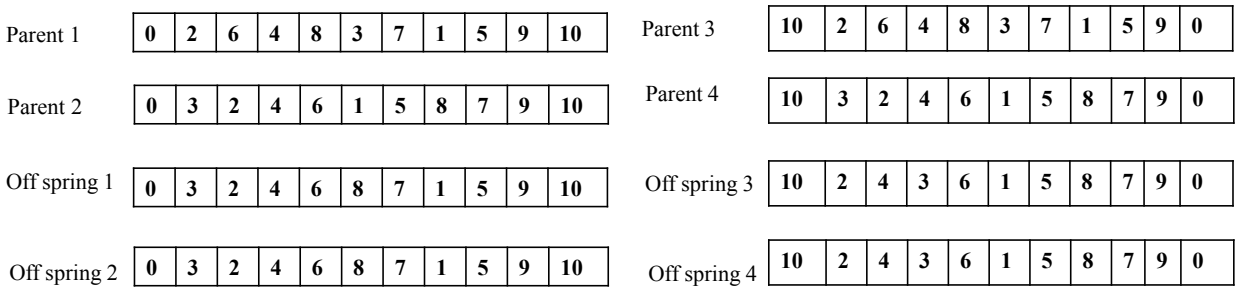

Figure 4.1: Crossover operation (using random crossover point) [8]

Algorithm 3 : To do crossover operation

1: Input the present population of chromosomes and compute the new population $P1$ and $P2$.
2: Take a logical value L=TRUE, Let $k = 0$, $M = P_{size} \times \frac{70}{100}$
3: **While** $(k \leq M)$ **do**
4: **If** $(\frac{M \times 30}{100} \leq 100)$ **then**
5: Create random number range from 0 to P_{size} and store result into $P1$
6: Create random number series excluding elite chromosomes range from 0 to P_{size} and store result into $P2$
7: **Else**
8: Create random number series excluding elite chromosomes from range 0 to $(P_{size} - 1)$ and store result into $P1$ and $P2$
9: **End If**
10: Apply crossover operator at crossover rate or crossover point i.
11: Display the two parents chromosomes lists into two sections i.e, left and right
12: Let offspring1= new offspring
13: Create relationship between $parent1$ and offspring1 (At left side)
14: Transfer the genes (non-available genes of left sections of $parent1$) of $parent2$ to the offspring1 (At right side)
15: **If** L==TRUE **then**
16: **If** $(random_{number} == 0)$ **then**
17: $P1$=offspring1
18: **Else**
19: $P1$ = offspring3
20: **End If**
21: **Else**
22: **If** $(rand_{number} == 0)$ **then**
23: $P1$ = offspring2
24: **Else**
25: $P1$ = offspring4
26: **End If**
27: **End While**
28: Display new population and stop processing

D. Mutation operator

There are two types of mutation in GA, naming immigration and allele type. In this algorithm, immigration mutation is used. For mutation process, a mutation operator is used to process a diversity in chromosome population with a defined mutation probability rate. Here, a particular chromosome is replaced randomly with another chromosome and evaluate the fitness function makespan ms till a better optimize result arrived. This process is shown in Figure 4.2. An iterative process of mutation will occur until a better optimal solution will be obtained. This process is illustrated in Algorithm 4.

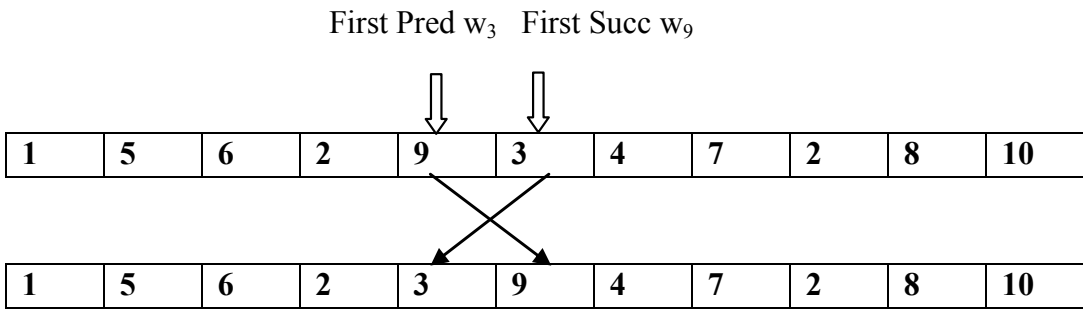

Figure 4.2: Mutation operator [9]

Algorithm 4 : Mutation operation process

1: Inputting the chromosome list of current population
2: A new population generated by applying mutation operator
3: Select the chromosome list randomly and Let $G = rand(gene)$
4: Select the first successor and Let $S = rand(gene)$ from chromosome list except for G
5: Offspring= interchanging G with S
6: Evaluate the makespan ms
7: **If** ms=minimum **then**
8: Terminate the process
9: **Else**
10: Go to the first step again (i.e. step 1)
11: **End If**
12: Print optimal result having makespan ms

E. Termination condition

Since it is not possible to achieve ms as 0, therefore, 100 function evaluations used to stop each section of the algorithm. The function evaluations mean the total number of time tried to calculate the ms, so-called fitness function or objective function.

Chapter 5

Performance analysis

Outline

This Chapter provides the details of experimental set-up and the results of all designed job-shop scheduling techniques. The comparisons have been drawn between proposed and well-known existing job-shop scheduling techniques. The mean improvements of proposed technique over others in percentage have also been evaluated by considering speedup, efficiency, utilization and mean gain time. Also, the mean reductions of proposed techniques over others in percentage (%) have also been evaluated by considering makespan, energy consumption, load imbalance rate and overhead time. Analysis of variance (ANOVA) based statistical testing has also been done to evaluate the significance of proposed techniques.

5.1 Experimental set-up

This section illustrates the experimental set-up and quantitative analysis of the proposed job-shop scheduling techniques. The proposed techniques have been tested on benchmark job-shop parallel problems. Different sizes of jobs and machines have been considered to evaluate the scalability effect of the proposed techniques. The proposed techniques have been compared with ACO based job-shop scheduling technique All job shop scheduling were executed on Intel core $i5$ machine @ 2.56 *GHz* with 6 GB RAM. MATLAB $2016a$ software is used in combination with parallel processing toolbox.

To successfully simulate the proposed techniques, MATLAB $2016a$ tool has been used. Also, to evaluate the effectiveness of the proposed techniques, the comparisons have also been drawn between the proposed and existing job-shop scheduling techniques. For efficiently comparing the proposed techniques, with existing job-shop scheduling

techniques. Job-shop benchmark parallel problems have been considered by taking the different sizes of jobs i.e., from 10 to 30.

As known in prior, efficient parameters selection provides efficient results. Therefore, the number of experiments have been performed by changing the values of parameter required for simulation of the proposed technique.

Table 5.1 depicts various parameters along with their tuned values.

Table 5.1: Simulation parameters

Parameter	Value	Technique
Population size	100	Genetic algorithm
Number of generations	100	Genetic algorithm
Selection operator	Randomization	Genetic algorithm
Crossover operator	Random one point	Genetic algorithm
Crossover rate	0.5	Genetic algorithm
Mutation rate	0.001	Genetic algorithm
Sampling rate	0.05	Genetic algorithm
Ants	5 to 10	Variable neighborhood search
alpha	0.2	Pheromone evaluation factor
beta	0.2	Pheromone evaluation factor
Number of machines	10 to 3	-

Each simulation is run at-least 30 times and min, $mean$ and max values of each performance measure have been taken for proposed and existing job-shop scheduling techniques. The goal behind taking these min, $mean$ and max values is to evaluate the resultant fluctuations that may occur due to random behavior of meta-heuristic techniques. Lesser variations in these values indicate that the technique(s) perform consistently well.

Table 5.2 and Figure 5.1 depict the comparison between existing and proposed technique in terms of average execution time (in seconds). It is found that the proposed technique has lesser average execution time as compared to ACO based job-shop scheduling technique.

Table 5.2: Average execution time analysis (in seconds)

Iteration	ACO	GACO
1	148.7022	122.7687
2	146.2933	107.8089
3	142.9511	118.2000
4	123.5067	107.5067
5	134.8444	116.6578
6	160.1333	135.4933
7	131.8222	108.0933
8	127.4400	112.9778
9	138.8933	123.0178
10	145.1022	112.1244

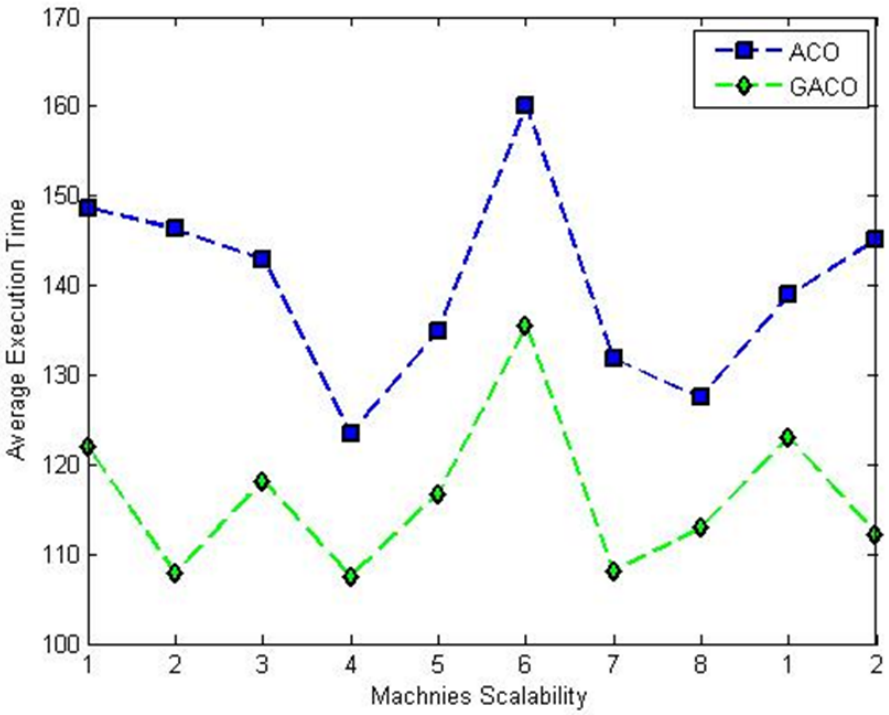

Figure 5.1: Average execution time analysis

Table 5.3 and Figure 5.6 demonstrate the comparison between existing and proposed technique in terms of makespan time (in seconds). It is found that the proposed technique has lesser makespan time as compared to ACO based job-shop scheduling technique.

Table 5.3: Makespan analysis (in seconds)

Iteration	ACO	GACO
1	1234	1114
2	1253	981
3	1075	958
4	1086	1016
5	1318	1210
6	1185	970
7	1055	983
8	1176	1110
9	1280	1006
10	1281	1147

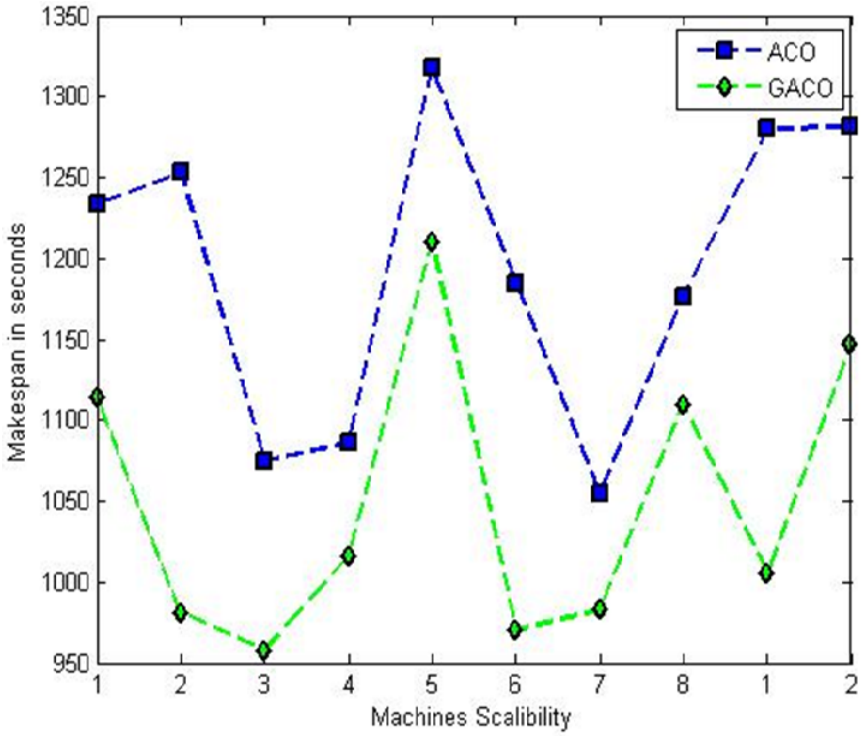

Figure 5.2: Makespan analysis of ACO and proposed technique

Reducing the energy consumed by parallel machines need to be minimized. It is preferable to achieve green computing environment. A job shop scheduling technique is said to be energy efficient if it consumes lesser energy. Table 5.4 and Figure 5.3 prove that the proposed technique consumes lesser energy in Joules (n_j) than the existing ACO based job-shop scheduling technique.

Table 5.4: Energy Consumption analysis (in joules)

Iteration	ACO	GACO
1	7.2064	5.5411
2	7.0123	6.0730
3	6.1526	5.446
4	6.3667	5.8139
5	7.6740	6.8724
6	6.7173	5.5014
7	6.1316	5.6267
8	6.7874	6.2853
9	7.2965	5.7059
10	7.4060	6.4531

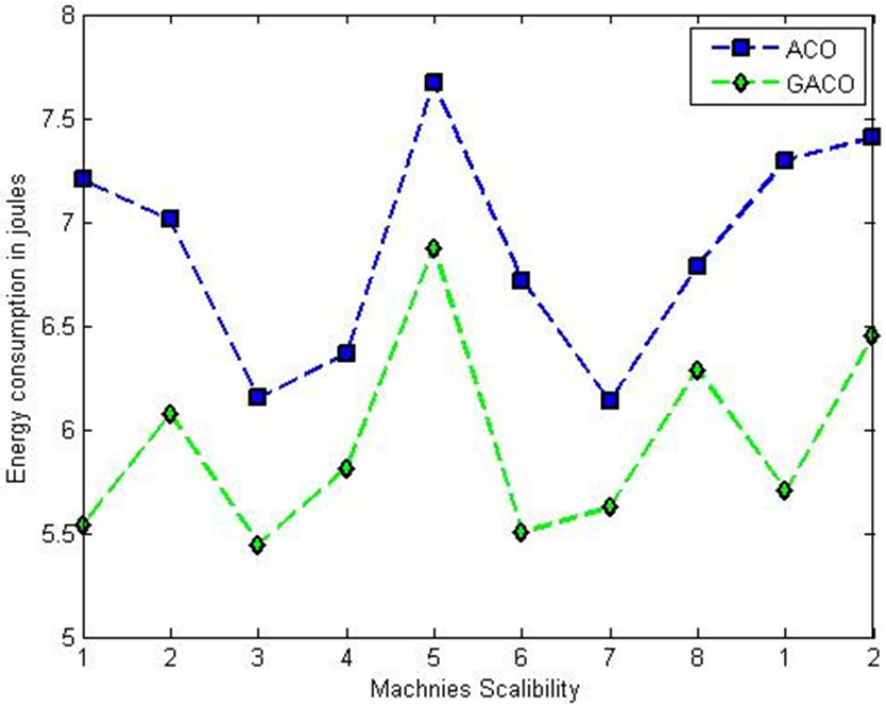

Figure 5.3: Energy consumption analysis in joules

Figure 5.4 and Table 5.5 show the comparison between existing and proposed technique by considering the average execution time in seconds. It has been observed that the GACO has significantly lesser average execution time as compared to existing ACO based job-shop scheduling technique. Also, GACO technique has shown lesser variability, therefore it performs consistently every time compared to existing scheduling techniques in terms of average execution time in seconds.

Table 5.5: Scalability analysis with respect to execution time

Iteration	Job shop size	Machines	ACO	GACO
1	15	15	2142	1647
2	20	15	2024	1611
3	20	20	2872	2415
4	30	15	2133	1778
5	30	20	3186	2710
6	50	15	1947	1523
7	50	20	2911	2433
8	100	20	2666	2572

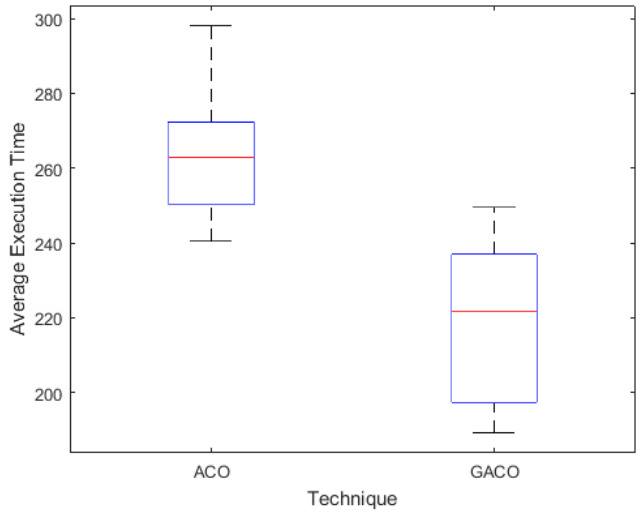

Figure 5.4: Boxplot analysis in terms of average execution time in joules

Figure 5.5 and Table 5.6 demonstrate the comparison between existing and proposed technique by considering the makespan (ms) time in seconds. It has been observed that the GACO has significantly lesser ms as compared to existing ACO based job-shop scheduling technique. Also, GACO technique has shown lesser variability, therefore it performs consistently every time compared to existing scheduling techniques.

Table 5.6: Scalability analysis with respect to makespan time

Iteration	Job shop size	Machines	ACO	GACO
1	15	15	12.5613	9.5940
2	20	15	11.7616	9.3590
3	20	20	16.8992	14.2121
4	30	15	12.4269	10.3475
5	30	20	18.7059	15.8303
6	50	15	11.3952	8.9311
7	50	20	17.1245	14.4272
8	100	20	15.7444	15.1137

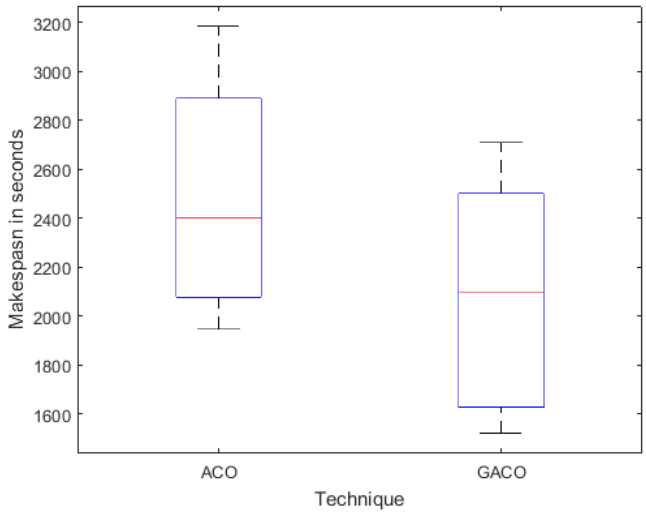

Figure 5.5: Boxplot analysis in terms of makespan time in seconds

Table 5.7 and Figure 5.6 show the comparative analysis of the energy consumption in joules between existing and the proposed technique. The proposed technique has lesser variability, therefore it achieves consistent energy consumption in every scale of jobs and machines. Also, GACO based job-shop scheduling technique has lesser energy consumption than ACO based job-shop scheduling techniques.

Table 5.7: Scalability analysis with respect to energy consumption in joules

Iteration	Job shop size	Machines	ACO	GACO
1	15	15	266.2222	200.4000
2	20	15	246.3733	194.3333
3	20	20	270.5625	227.6025
4	30	15	259.6578	215.7022
5	30	20	298.0950	249.5150
6	50	15	240.4800	189.2756
7	50	20	274.0250	232.7575
8	100	20	254.0175	241.2825

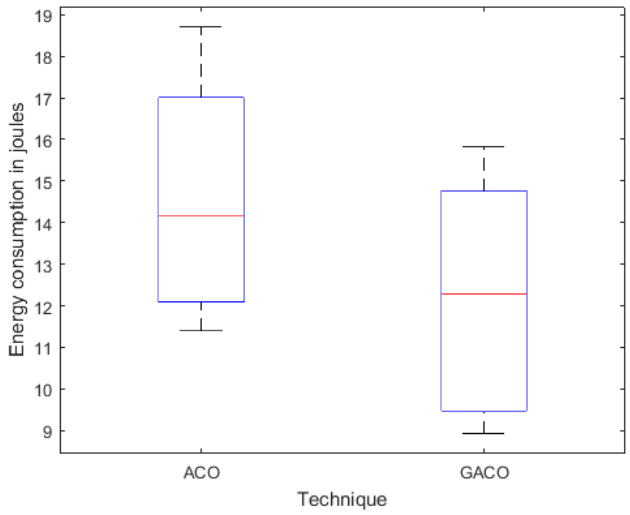

Figure 5.6: Boxplot analysis in terms of energy consumption in joules

After evaluating the performance of meta-heuristic based job-shop scheduling techniques, it has been concluded that the proposed technique is most suitable for optimizing the makespan, energy consumption and average execution time, respectively.

Chapter 6

Conclusion and future work

6.1 Conclusion

The aim of this research work has been to design and develop efficient job-shop scheduling techniques using hybrid meta-heuristic techniques for parallel machines. It has been addressed by the proposed hybrid Ant colony optimization (ACO) and Genetic algorithm (GA). As studied, the efficient meta-heuristic based job-shop scheduling techniques are the important techniques in bringing down the makespan and energy consumption rate in parallel machines. However, the majority of the existing meta-heuristic techniques suffer from one of these issues: premature convergence, poor convergence speed, initially selected random solutions and stuck in local optima.

Therefore, to overcome above-mentioned issues, in this research work hybrid meta-heuristics based job-shop scheduling technique has been proposed. The proposed job-shop scheduling technique has integrated ACO with GA. To implement the proposed techniques, a step by step methodology has been utilized. Initially, a machines based model is designed by considering the well-known job-shop benchmark problems. MATLAB $2016a$ tool has been used to successfully simulate the proposed techniques.

To evaluate the effectiveness of the proposed techniques, extensive experiments have been done in this research work. The proposed technique has been compared with well-known ACO based job-shop scheduling technique. It has been concluded that the proposed technique has reduced the average execution time, makespan and energy consumption by 2.41%, 1.79% and 2.18%, respectively than ACO based job-shop scheduling. By reducing consumed energy, all the proposed techniques, indirectly reduced carbon emissions and cooling requirements of the machines data centers, leading to a further

drop in the energy demand and helping in achieving green computing.

The precise contributions of this research work are listed below:

1. An extensive literature survey has been done to study existing meta-heuristic based job-shop scheduling techniques.

2. A novel hybrid meta-heuristic technique is designed.

3. The proposed technique has been implemented in the MATLAB $2016a$ software to successfully simulate the proposed technique.

5. The obtained results have been verified quantitatively by using the average execution time, makespan time and energy consumption in joules, to evaluate the significant improvement of the proposed technique over others.

6.2 Future work

The hybrid meta-heuristic based job-shop scheduling technique can be enhanced for future research. Some of the research directions are outlined below:

1. It would be exciting to examine the possible integration of the proposed technique with various other Quality of services (QoS) parameters such as cost, fault tolerance, and reliability *etc*.

2. The proposed technique has not addressed the communication overhead issue in parallel machines. As a part of a future research, these issues will also be considered as significant challenges in a business-driven parallel machine.

3. Besides, all the proposed technique can be enhanced to support disk-intensive and network workloads along with the network and the storage resources and can be tested with an increased data set and more number of machines to achieve an enhanced efficiency.

Bibliography

[1] S. Nguyen, M. Zhang, M. Johnston, and K. C. Tan, "A computational study of representations in genetic programming to evolve dispatching rules for the job shop scheduling problem," *IEEE Transactions on Evolutionary Computation*, vol. 17, pp. 621–639, Oct 2013.

[2] K. Rameshkumar and C. Rajendran, "A novel discrete pso algorithm for solving job shop scheduling problem to minimize makespan," in *IOP Conference Series: Materials Science and Engineering*, vol. 310, p. 012143, IOP Publishing, 2018.

[3] M. Nouiri, A. Bekrar, A. Jemai, S. Niar, and A. C. Ammari, "An effective and distributed particle swarm optimization algorithm for flexible job-shop scheduling problem," *Journal of Intelligent Manufacturing*, vol. 29, no. 3, pp. 603–615, 2018.

[4] S. Sivasubramani and K. Swarup, "Environmental/economic dispatch using multiobjective harmony search algorithm," *Electric power systems research*, vol. 81, no. 9, pp. 1778–1785, 2011.

[5] A. Muthiah, A. Rajkumar, and R. Rajkumar, "Hybridization of artificial bee colony algorithm with particle swarm optimization algorithm for flexible job shop scheduling," in *Energy Efficient Technologies for Sustainability (ICEETS), 2016 International Conference on*, pp. 896–903, IEEE, 2016.

[6] A. Ouaarab, B. Ahiod, X.-S. Yang, and M. Abbad, "Discrete cuckoo search algorithm for job shop scheduling problem," in *Intelligent Control (ISIC), 2014 IEEE International Symposium on*, pp. 1872–1876, IEEE, 2014.

[7] T.-K. Dao, T.-S. Pan, J.-S. Pan, *et al.*, "Parallel bat algorithm for optimizing makespan in job shop scheduling problems," *Journal of Intelligent Manufacturing*, pp. 1–12, 2015.

[8] J. C. Chen, C.-C. Wu, C.-W. Chen, and K.-H. Chen, "Flexible job shop scheduling with parallel machines using genetic algorithm and grouping genetic algorithm," *Expert Systems with Applications*, vol. 39, no. 11, pp. 10016–10021, 2012.

[9] J. Kaabi, C. Varnier, and N. Zerhouni, "Genetic algorithm for scheduling production and maintenance in a flow shop," in *Proceedings of the IMS04 Conference, Arles, France*, 2003.

[10] E. Taillard, "Benchmarks for basic scheduling problems," *http://mistic.heig-vd.ch/taillard/problemes.dir*, 2018.

[11] S. Balin, "Parallel machine scheduling with fuzzy processing times using a robust genetic algorithm and simulation," *Information Sciences*, vol. 181, no. 17, pp. 3551–3569, 2011.

[12] K. P. Adzakpa, K. H. Adjallah, and F. Yalaoui, "On-line maintenance job scheduling and assignment to resources in distributed systems by heuristic-based optimization," *Journal of intelligent manufacturing*, vol. 15, no. 2, pp. 131–140, 2004.

[13] S. Ponnambalam, V. Ramkumar, and N. Jawahar, "A multiobjective genetic algorithm for job shop scheduling," *Production planning & control*, vol. 12, no. 8, pp. 764–774, 2001.

[14] T. Borreguero-SanchidriÃin, R. Pulido, . GarcÃa-SÃinchez, and M. Ortega-Mier, "Flexible job shop scheduling with operators in aeronautical manufacturing: A case study," *IEEE Access*, vol. 6, pp. 224–233, 2018.

[15] T. Jamrus, C. F. Chien, M. Gen, and K. Sethanan, "Hybrid particle swarm optimization combined with genetic operators for flexible job-shop scheduling under uncertain processing time for semiconductor manufacturing," *IEEE Transactions on Semiconductor Manufacturing*, vol. 31, no. 1, pp. 32–41, 2018.

[16] Y. Mei, S. Nguyen, B. Xue, and M. Zhang, "An efficient feature selection algorithm for evolving job shop scheduling rules with genetic programming," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 1, no. 5, pp. 339–353, 2017.

[17] H. C. Chang, Y. P. Chen, T. K. Liu, and J. H. Chou, "Solving the flexible job shop scheduling problem with makespan optimization by using a hybrid taguchi-genetic algorithm," *IEEE Access*, vol. 3, pp. 1740–1754, 2015.

[18] I. Kacem, S. Hammadi, and P. Borne, "Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 32, no. 1, pp. 1–13, 2002.

[19] D. Lei, "Pareto archive particle swarm optimization for multi-objective fuzzy job shop scheduling problems," *The International Journal of Advanced Manufacturing Technology*, vol. 37, no. 1-2, pp. 157–165, 2008.

[20] M. A. Abido, "Environmental/economic power dispatch using multiobjective evolutionary algorithms," *IEEE transactions on power systems*, vol. 18, no. 4, pp. 1529–1537, 2003.

[21] L. Asadzadeh, "A local search genetic algorithm for the job shop scheduling problem with intelligent agents," *Computers & Industrial Engineering*, vol. 85, pp. 376–383, 2015.

[22] F. Zhao, J. Tang, J. Wang, *et al.*, "An improved particle swarm optimization with decline disturbance index (ddpso) for multi-objective job-shop scheduling problem," *Computers & Operations Research*, vol. 45, pp. 38–50, 2014.

[23] L. Wang, Q.-K. Pan, and M. F. Tasgetiren, "A hybrid harmony search algorithm for the blocking permutation flow shop scheduling problem," *Computers & Industrial Engineering*, vol. 61, no. 1, pp. 76–83, 2011.

[24] J. Harbering, A. Ranade, M. Schmidt, and O. Sinnen, "Complexity, bounds and dynamic programming algorithms for single track train scheduling," *Annals of Operations Research*, pp. 1–22, 2017.

[25] C. O. Kim and H. J. Shin, "Scheduling jobs on parallel machines: a restricted tabu search approach," *The International Journal of Advanced Manufacturing Technology*, vol. 22, no. 3-4, pp. 278–287, 2003.

[26] J. E. Schaller, "Minimizing total tardiness for scheduling identical parallel machines with family setups," *Computers & Industrial Engineering*, vol. 72, pp. 274–281, 2014.

[27] X. Li and L. Gao, "An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem," *International Journal of Production Economics*, vol. 174, pp. 93–110, 2016.

[28] J. Kuhpfahl and C. Bierwirth, "A study on local search neighborhoods for the job shop scheduling problem with total weighted tardiness objective," *Computers & Operations Research*, vol. 66, pp. 44–57, 2016.

[29] S. Melouk, P. Damodaran, and P.-Y. Chang, "Minimizing makespan for single machine batch processing with non-identical job sizes using simulated annealing," *International journal of production economics*, vol. 87, no. 2, pp. 141–147, 2004.

[30] W.-C. Lee, C.-C. Wu, and P. Chen, "A simulated annealing approach to makespan minimization on identical parallel machines," *The International Journal of Advanced Manufacturing Technology*, vol. 31, no. 3-4, pp. 328–334, 2006.

[31] P. Damodaran and M. C. Vélez-Gallego, "A simulated annealing algorithm to minimize makespan of parallel batch processing machines with unequal job ready times," *Expert Systems with Applications*, vol. 39, no. 1, pp. 1451–1458, 2012.

[32] W.-C. Yeh, P.-J. Lai, W.-C. Lee, and M.-C. Chuang, "Parallel-machine scheduling to minimize makespan with fuzzy processing times and learning effects," *Information Sciences*, vol. 269, pp. 142–158, 2014.

[33] N. Shivasankaran, P. S. Kumar, and K. V. Raja, "Hybrid sorting immune simulated annealing algorithm for flexible job shop scheduling," *International Journal of Computational Intelligence Systems*, vol. 8, no. 3, pp. 455–466, 2015.

[34] T.-K. Dao, T.-S. Pan, J.-S. Pan, *et al.*, "Parallel bat algorithm for optimizing makespan in job shop scheduling problems," *Journal of Intelligent Manufacturing*, vol. 29, no. 2, pp. 451–462, 2018.

[35] S. Rajakumar, V. Arunachalam, and V. Selladurai, "Workflow balancing in parallel machines through genetic algorithm," *The International Journal of Advanced Manufacturing Technology*, vol. 33, no. 11-12, pp. 1212–1221, 2007.

[36] L.-Y. Tseng and Y.-T. Lin, "A hybrid genetic local search algorithm for the permutation flowshop scheduling problem," *European Journal of Operational Research*, vol. 198, no. 1, pp. 84–92, 2009.

[37] I. A. Chaudhry and P. R. Drake, "Minimizing total tardiness for the machine scheduling and worker assignment problems in identical parallel machines using genetic algorithms," *The International Journal of Advanced Manufacturing Technology*, vol. 42, no. 5-6, p. 581, 2009.

[38] R. Tavakkoli-Moghaddam, F. Taheri, M. Bazzazi, M. Izadi, and F. Sassani, "Design of a genetic algorithm for bi-objective unrelated parallel machines scheduling with sequence-dependent setup times and precedence constraints," *Computers & Operations Research*, vol. 36, no. 12, pp. 3224–3230, 2009.

[39] L. Zhang, L. Gao, and X. Li, "A hybrid genetic algorithm and tabu search for a multi-objective dynamic job shop scheduling problem," *International Journal of Production Research*, vol. 51, no. 12, pp. 3516–3531, 2013.

[40] A. C. Spanos, S. T. Ponis, I. P. Tatsiopoulos, I. T. Christou, and E. Rokou, "A new hybrid parallel genetic algorithm for the job-shop scheduling problem," *International Transactions in Operational Research*, vol. 21, no. 3, pp. 479–499, 2014.

[41] R. Zhang and R. Chiong, "Solving the energy-efficient job shop scheduling problem: a multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption," *Journal of Cleaner Production*, vol. 112, pp. 3361–3375, 2016.

[42] C. Peng, Y. Fang, P. Lou, and J. Yan, "Analysis of double-resource flexible job shop scheduling problem based on genetic algorithm," in *Networking, Sensing and Control (ICNSC), 2018 IEEE 15th International Conference on*, pp. 1–6, IEEE, 2018.

[43] S. A. Torabi, N. Sahebjamnia, S. A. Mansouri, and M. A. Bajestani, "A particle swarm optimization for a fuzzy multi-objective unrelated parallel machines scheduling problem," *Applied Soft Computing*, vol. 13, no. 12, pp. 4750–4762, 2013.

[44] L. Yin, L. Yang, and M. Hu, "Job shop scheduling based on improved discrete particle swarm optimization," in *Proceedings of the 21st international conference on industrial engineering and engineering management 2014*, pp. 99–101, Springer, 2015.

[45] C.-C. Wu, J.-Y. Chen, W.-C. Lin, K. Lai, S.-C. Liu, and P.-W. Yu, "A two-stage three-machine assembly flow shop scheduling with learning consideration to minimize the flowtime by six hybrids of particle swarm optimization," *Swarm and Evolutionary Computation*, 2018.

[46] R. Ruiz, J. C. García-Díaz, and C. Maroto, "Considering scheduling and preventive maintenance in the flowshop sequencing problem," *Computers & Operations Research*, vol. 34, no. 11, pp. 3314–3330, 2007.

[47] Y.-C. Liang, Y.-M. Hsiao, and C.-Y. Tien, "Metaheuristics for drilling operation scheduling in taiwan pcb industries," *International Journal of Production Economics*, vol. 141, no. 1, pp. 189–198, 2013.

[48] M. F. Zarandi and V. Kayvanfar, "A bi-objective identical parallel machine scheduling problem with controllable processing times: a just-in-time approach," *The International Journal of Advanced Manufacturing Technology*, vol. 77, no. 1-4, pp. 545–563, 2015.

[49] B. Zhao, J. Gao, K. Chen, and K. Guo, "Two-generation pareto ant colony algorithm for multi-objective job shop scheduling problem with alternative pro-

cess plans and unrelated parallel machines," *Journal of Intelligent Manufacturing*, vol. 29, no. 1, pp. 93–108, 2018.

[50] K. Udaiyakumar and M. Chandrasekaran, "Application of firefly algorithm in job shop scheduling problem for minimization of makespan," *Procedia Engineering*, vol. 97, pp. 1798–1807, 2014.

[51] A. Berrichi, F. Yalaoui, L. Amodeo, and M. Mezghiche, "Bi-objective ant colony optimization approach to optimize production and maintenance scheduling," *Computers & Operations Research*, vol. 37, no. 9, pp. 1584–1596, 2010.

ORIGINALITY REPORT

16%	7%	14%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Yuvraj Gajpal. "An ant colony algorithm for scheduling in flowshops with sequence-dependent setup times of jobs", The International Journal of Advanced Manufacturing Technology, 09/2006 Publication	2%
2	Atabak Elmi, Seyda Topaloglu. "Cyclic job shop robotic cell scheduling problem: Ant colony optimization", Computers & Industrial Engineering, 2017 Publication	1%
3	Simranpreet Kaur, Shivani Sharma. "Performance Evaluation of Artificial Bee Colony and Compressive Sensing Based Energy Efficient Protocol for WSNs", 2017 13th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), 2017 Publication	1%
4	ikucukkoc.baun.edu.tr Internet Source	1%

Scanned by CamScanner