Computer Engineering and Intelligent Systems
ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online)
Vol.11, No.2, 2020

An Enhanced Model for Job Sequencing and Dispatch in Identical Parallel Machines

Uzochukwu Onwuachu[1*], C. Ugwu[2] and Edem Williams [3]

1. Department of Computer Science, Imo State University, Owerri, Imo State, Nigeria.
2. Department of Computer Sciences, University of Port Harcourt, Choba, Rivers State, Nigeria.
3. Department of Computer Science, University of Calabar, Calabar, Cross River State, Nigeria.

*E-mail of the corresponding author: uzochukwuchristian@yahoo.com

Abstract
This paper has developed an efficient scheduling model that is robust and minimizes the total completion time for job completion in identical parallel machines. The new model employs Genetic-Fuzzy technique for job sequencing and dispatch in identical parallel machines. It uses genetic algorithm technique to develop a job scheduler that does the job sequencing and optimization while fuzzy logic technique was used to develop a job dispatcher that dispatches job to the identical parallel machines. The methodology used for the design is the Object Oriented Analysis and Design Methodology (OOADM) and the system was implemented using C# and .NET framework. The model was tested with fifteen identical parallel machines used for printing. The parameters used in analyzing this model include the job scheduling length, average execution time, load balancing and machines utilization. The result generated from the developed model was compare with the result of other job scheduling models like First Come First Sever (FCFS) scheduling approach and Genetic Model (GA) scheduling approach. The result of the new model shows a better load balancing and high machine utilization among the individual machines when compared with the First Come First Serve (FCFS) scheduling model and Genetic Algorithm (GA) scheduling model.

Keywords: Parallel Machines, Genetic Model, Job Scheduler, Fuzzy Logic Technique, Load Balancing, Machines Utilization
DOI: 10.7176/CEIS/11-2-05
Publication date: March 31st 2020

I. Introduction

Scheduling is an important issue in any type of system (Ramamritham and Stankovic,1994). Job scheduling comprises of arranging the request of execution of job with the goal that the planning constraints are met. Schedule is viewed as great if the planning constraints of the considerable number of jobs are met. The challenges of job scheduling in a parallel machine are dealing with limited computing resources for the number of jobs, considering the following factors which include, resource starvation, load balancing complexity, dependency and efficiency (Rachhpal, 2016). Scheduling decides the job allocation and job grouping at each machine. Utilizing optimization and dispatching rules are two well-known techniques for taking care of scheduling issues. An optimization scans the arrangement space for the best job sequencing; this is performed all inclusive for all machines (Rachhpal, 2014). The second strategy to take care of a scheduling issue is by utilizing dispatching rules to figure out what job to process straightaway. The scheduling problem works with the optimal assignment of jobs to the parallel and orders their execution so that the total completion time is reduced (Jasbir and Gurvinder, 2012).

Parallel systems are resources that are shared by communities of users. Job scheduling will decide when and how each job should be carried out in order to make maximum use of the available machine (Weinberg, 2002). The way jobs are allocated to machine is fundamental to achieving high performance within the parallel systems, these includes minimizing the job response time and maximizing system throughput (Neelu and Sampada, 2012). The aim of job scheduling has to do with achieving load balancing among the identical parallel systems and minimization of overall execution time of all the parallel system (Karthick, 2011). Scheduling can exploit the system's throughput by completing maximum number of jobs within a given time. Load balancing can be an issue when jobs are scheduled with limited resources, and should be minimized in other to improve the system throughput and efficiency (Lei et al., 2006)

Abraham and Braun (2000) proposed three basic heuristics technique for Grid scheduling, these includes Genetic Algorithm (Song et al., 2005), Simulated Annealing (Orosz and Jacobson, 2002) and Tabu Search (Braun et al., 2001). Genetic Algorithm is a powerful optimization technique which is inspired by nature and it is simulated from the evolutionary natural selection process. The improved solution of generation is estimated with the fitness value and the individual with better fitness values are used to generate more solutions using crossover technique and mutation processes. During the time of optimization, some machine may be left alone without allocating any jobs to them; this can be referred as the idle time of the machine (Rachhpal, 2012)

The job scheduling algorithm is considered a complex process because it must schedule a large number of jobs into the available resources. (Rajakumar et al., 2006; Safwat and Fatma 2016). This paper identifies a number of performance issues that may be experienced when utilizing the identical parallel machine in the scenario of job scheduling, namely mapping, arrangement of execution, and optimal configuration of the identical parallel machines. There is a need therefore, to develop a model for job sequencing and dispatch to conquer these issues (Rachhpal, 2014). The enabling strategy is the use of job scheduling plan based on Genetic Algorithm and fuzzy logic to determine scheduling issues (Ramkumar et al., 2011)

II Literature Review

Xiaofeng and Yuesheng (2010) worked on world optimization primarily based on hybrid clone decision genetic algorithm for undertaking scheduling. They introduced a optimization mannequin for the undertaking scheduling problem and developed a hybrid clone resolution genetic algorithm (HCSGA) to effectively resolve it. The analysis and test outcomes confirmed that HCSGA has the characteristics of speedy convergence, exact international search capacity, and is highest quality to different algorithms simultaneously.

Liang et. al. (2010) labored on solving job keep scheduling hassle the usage of genetic algorithm with penalty function. They existing a genetic algorithm with a penalty function for the job store scheduling problem. A clone selection based hyper mutation and a lifespan prolonged approach used to be designed and an adaptive penalty characteristic used to be designed so that the algorithm can search in both viable and unfeasible areas of the solution space. Simulated experiments were performed on 23 benchmark instances taken from the OR-library. The outcomes showed the effectiveness of the proposed algorithm.

Kamaljit et. al. (2010) labored on heuristics based totally genetic algorithm for scheduling static tasks in homogeneous parallel system. The trouble of same execution time or completion time and equal precedence in the homogeneous parallel machine was once resolved by using the use of the notion of Bottom-level (b-level) or Top-level (t-level). The blended method named as heuristics based totally genetic algorithm (HGA) based on MET (Minimum Execution Time)/Min-Min heuristics and b-level or t-level precedence decision and used to be in contrast with a pure genetic algorithm, min-min heuristic, MET heuristic and First Come First Serve (FCFS) approach. The consequences of their experiments confirmed that the heuristics based totally genetic algorithm produces plenty higher consequences in terms of high-quality of solutions.
Moghaddam et. al. (2010) worked on Fuzzy Multi-Objective Linear Programming (FMOLP) mannequin for fixing a multi-objective single-machine scheduling problem. Their proposed model tries to decrease the total weighted tardiness and makespan simultaneously.

Ye and Yan (2010) worked on a genetic algorithm for job-shop scheduling in which they analyzed the characteristics of the job store scheduling problem. A new genetic algorithm for fixing the agile job store scheduling used to be introduced to remedy the job save scheduling problem. The relevant crossover and mutation operation was additionally designed. It jumped from the local superior solution, and the search place of solution was improved. The result confirmed that the genetic algorithm has been correctly applied to the job shop scheduling problems.

Antonio et. al. (2011) labored on a multi-criteria meta-fuzzy-scheduler for impartial tasks in grid computing. They used Meta-Scheduler for grid computing which does now not want any given records about tasks size or tasks arrival time, unlike traditional dynamic heuristics. Experimental outcomes using fuzzy scheduler confirmed that thru their proposal, they done these two goals and improved dynamic heuristics in prior literature.

Mostafa and Medhat (2011) labored on hybrid algorithm for multiprocessor mission scheduling. They proposed three distinctive representations for the chromosomes of genetic algorithm: Task List (TL), Processor List (PL) and mixture of each (TLPLC). The done results confirmed that the proposed procedures significantly surpassed the other methods in terms of task execution time (makespan) and processor efficiency.

Tufan et. al. (2011) worked on a genetic algorithm strategy for minimizing complete tardiness in parallel computing device scheduling problems. Their learn about look into parallel machine scheduling hassle in order to limit complete tardiness and increase a genetic algorithm solution technique for such problems. It confirmed an environment friendly answer improvement scheme and an excellent crossover operator which was developed and built-in into the genetic algorithm.

Savas (2011) labored on non-identical parallel computer scheduling using genetic algorithm. He proposed a new crossover operator and optimality criterion in order to adapt the GA to non-identical parallel laptop scheduling problem. The algorithm was once tested on a numerical example by implementing it in simulation software and computational consequences were in contrast to those obtained with Longest Processing Time dispatching rule. The result of his lookup used to be promising. The findings confirmed that, in addition to its high computational speed for larger scale problem, the GA proposed here fits the non-identical parallel machine scheduling trouble of minimizing the most completion time (makespan).

Turker two and Sel (2011) worked on scheduling two parallel machines with sequence-dependent setups and a single server. The scheduling hassle on parallel machines with sequence-dependent setup times and setup operations that were performed by means of a single server. Their purpose was to get minimal makespan of the schedule. Their gadget was formulated as genetic algorithm with hassle sizes consisting of two machines and 10, 20 and 30 jobs. The choicest outcomes had been received using a string based permutation algorithm which scanned all alternatives.

Jaspal and Harsharanpal, (2011) worked on efficient tasks scheduling for heterogeneous multiprocessor the use of genetic algorithm with node duplication. The scheduling problem regarded in their work was bringing out the most suitable mapping of duties and their efficiently viable execution stream on multiprocessor system configuration. Several options and heuristics have been proposed to remedy this problem. They exhibited affectivity of Node duplication GA-based method by way of comparing towards some of the current deterministic scheduling methods for minimizing inter-processor site visitors' communication.

Fayçal et. al. (2013) worked on genetic algorithm for the parallel desktop scheduling trouble with consumable resources. Their research targeted on the scheduling hassle on equal parallel machines with consumable sources in order to limit the makespan criterion. Each job fed on countless aspects which arrived at one of a kind times. The arrival of every thing was represented with the aid of a curve-shaped staircase. A genetic algorithm was proposed to remedy this problem due to established brilliant performance in fixing combinatorial optimization problems. The computation consequences confirmed that this algorithm outperforms heuristic system and used to be tailor-made for larger scale problems.

Sachi et al (2013) worked on an Efficient and Robust Genetic Algorithm for Multiprocessor Task Scheduling, In their work, they developed a genetic algorithm based totally on the standards of evolution found in nature for discovering an most appropriate solution. Genetic algorithm is primarily based on three operators, which consist of the Natural Selection, Crossover and Mutation. To compare the overall performance of their algorithm, they additionally carried out some other scheduling algorithm HEFT, which is a heuristic algorithm. Simulation outcomes comprised of three parts, that is, the pleasant of solutions, robustness of genetic algorithm, and effect of mutation probability on overall performance of genetic algorithm.

Selvi. (2014) worked on multi-objective optimization problems on identical parallel machine scheduling using genetic algorithms. Their research tried to solve scheduling issues involving same parallel machines, the place the goal was once to optimize the multi-objective scheduling issues the usage of Genetic algorithms. The predominant contribution of the current work lies in proposing mathematical models using Genetic algorithms to optimize main goals and to study the effectiveness of these algorithms for small and large size problems.

Rajendra and Rakesh (2015) worked on adaptive job scheduling on computational grid for useful resource allocation and job completion Based on Feature Selection, for the enhancement of job decision in multi-criteria ant colony optimization used TLBO determination process. The TLBO is new meta-heuristic function. The modified procedure selection algorithm in ant colony optimization anticipated better job for the constrained useful resource allocation process. They modified ACO-TLBO aid allocation along with jobs simulated in MATLAB software. MATLAB is properly recognised simulation software program for the algorithm analysis. The experimental process used a unique size of grid. Their experimental end result shows that better job completion ratio instead of MC-ACO.

Fuqing et. al. (2015) worked on an improved shuffled complex evolution algorithm with sequence mapping mechanism for job keep scheduling problems. The sequence mapping mechanism was once used to change the variables in the non-stop domain to discrete variables in the combinational optimization problem; the sequence, which is based totally on job permutation, was once adopted for encoding mechanism and sequence insertion mechanism for decoding. While considering that the primary SCE algorithm has the drawbacks of bad answer and decrease price of convergence, a new strategy is used to trade the individual's evolution in the simple SCE algorithm. The method makes the new character closer to nice character in the cutting-edge population. The accelerated SCE algorithm (ISCE) was once used to resolve the regular job save troubles and the consequences confirmed that the improved algorithm was positive to the job store scheduling.

Marish and Amit , (2015) looked at reliable and efficient task scheduling algorithm based totally on genetic algorithm in cloud computing environment. Their algorithm was once applied and built-in using genetic algorithm for optimization of the results, including the value and performance factor. The device was once producing efficient outcomes in phrases of the most reliable solution when execution the usage of genetic algorithm.. The proposed system offers higher effects in completed the usage of genetic algorithm that is one of the outstanding metaheuristic techniques.

Rachhpal (2016) optimized task duplication based scheduling in parallel system. He discussed about how the parallel computation is carried out, and severa overall performance issues are also located as an open issue. The threat encountered in parallel computing used to be the motivation to analyze distinct optimization methods to accomplish the tasks barring volatile environment. Genetic Algorithm (GA) is some other method to make the concept of scheduling convenient and fast. Their work presented a Task Duplication-based Genetic Algorithm with Load Balance (TD-GA) approach on parallel processing for advantageous scheduling of a couple of duties with much less agenda length and load balance. TD-GA algorithm really dealt with the issues very nicely and the consequences confirmed that complexity, load balance and aid utilization are finely managed when in contrast to the other optimization approaches.

Thamilselvan and Natesan (2016) labored on hybridization of genetic algorithm with cuckoo search algorithm for job scheduling on distributed heterogeneous computing systems. They evaluated the proposed hybrid algorithm the use of one-of-a-kind Grid eventualities generated by a Grid simulator. The technique efficiently scheduled jobs onto reachable assets and result showed that the scheduler outperformed existing implementation in a grid environment, consequently ensuing to and additionally revealing their affectivity when makespan and flow time are minimized.

III **Materials and Methods**

The new system uses genetic-fuzzy model for job sequencing and dispatch in identical parallel machines. The schedules are generated in a manner that the chromosome is feasible after performing genetic operation. The system uses genetic model for job sequencing base on better fitness (job processing time and job due date) and fuzzy logic system to distribute the jobs based on job waiting time and the availability of the machines. The two fitness functions applied to evaluate the solution were based on the longest processing time and earliest due date of the jobs. The optimal solution is the one that maximizes the fitness function.

The job dispatcher takes the output of the genetic process as an input and performs a fuzzy reasoning on it, considering the job waiting time and machine availability in order to know which job to be dispatched to the individual machines. The fuzzification module receives the crisp numeric values as input, approach them and map them into fuzzy membership characteristic values. The fuzzy engine is in charge for processing all decided membership attribute values the use of fuzzy sets', with fuzzy rule base to pick out the most terrific fuzzy output. However, the defuzzification module is accountable for converting the fuzzy output into a numeric output fabulous for the environment selection and manage situation. The fuzzy inference machine determines the membership feature for the job waiting time and machine availability which will be used in job allocation to the same parallel systems.. The fuzzy sets for job waiting time depends on how long a particular job has to wait to be dispatched , the waiting time fuzzy values are given as Short, Average, and Long. The machine availability depends on the state of the processing machines; machine availability fuzzy values are given as free, busy and failed.

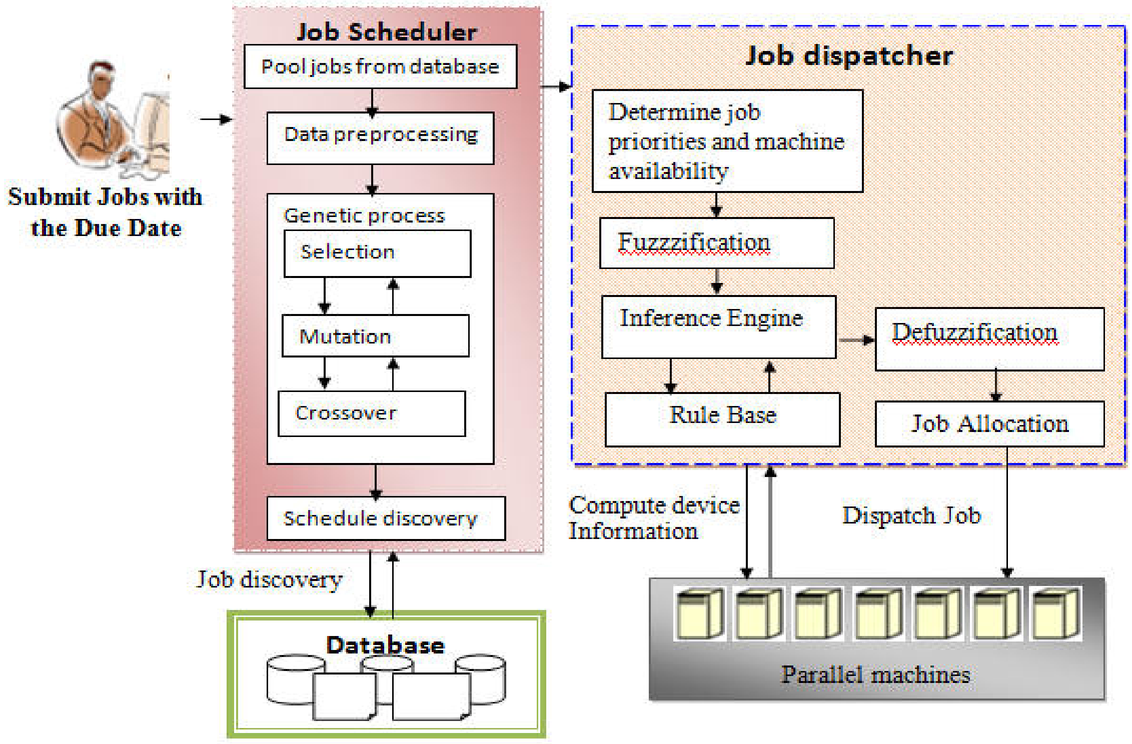


Figure 1: Architecture of the Proposed System.

`

Figure 1. shows the proposed system architecture. The proposed scheduling model process considers various component combined together to provide an optimized and efficient scheduling system. These components includes the

Job Scheduler: When jobs are submitted to the system, the scheduler stores the jobs directly in the cloud**.** The scheduler will periodically pull the jobs to be scheduled from the cloud and execute some pre-processing activities. The job scheduler uses genetic model for the job sequencing. Its fattiness function considers the longest processing time and the earliest due date. The job scheduler was developed with genetic model to sequence jobs for the job dispatcher.

Job Dispatcher: The job dispatcher collects job from the dispatching queue and determines the job waiting time and the machine availability using the fuzzy logic technique. Based on the determined waiting and machine availability, the dispatcher determines the job that goes to the identical parallel machine. The dispatcher also computes the resources information to know the available resources before transferring jobs to the machines for processing.

The Servers (Data as a Service): The proposed scheduling system uses a web server for storage of jobs. Once the user submits the job the scheduler stores the job in the server and periodically pulls the job for scheduling and allocation.

Parallel machines. The identical parallel machines execute the allocated jobs. Before a job is allocated to a machine, the job dispatcher will first ascertain the availability of the machine.

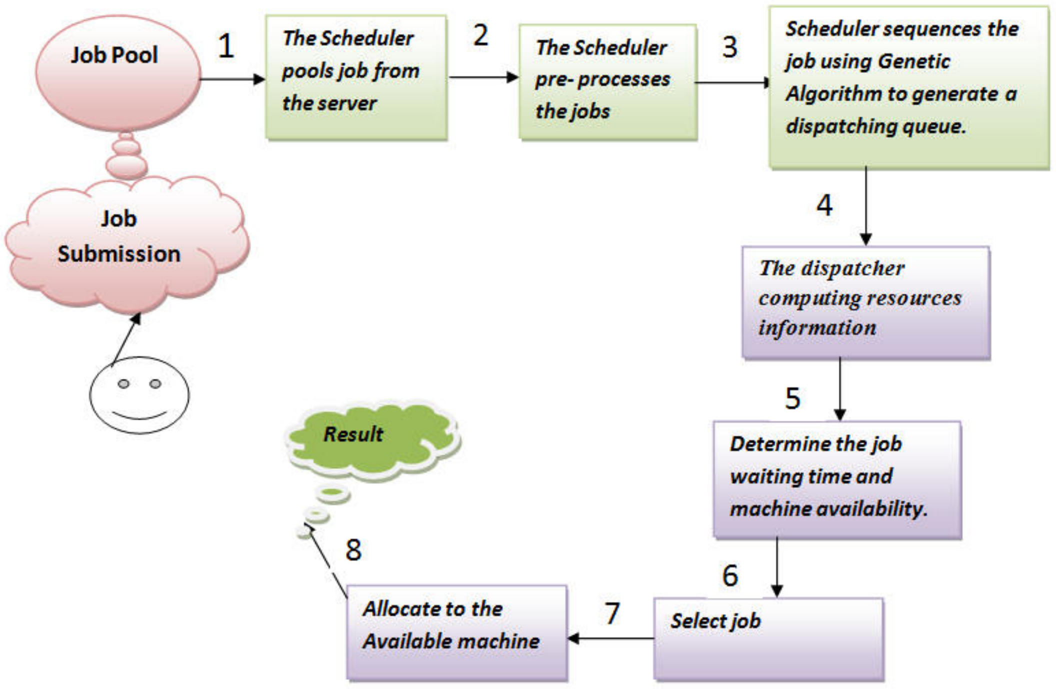

Job Pool
1
The Scheduler pools job from the server
2
The Scheduler pre- processes the jobs
3
Scheduler sequences the job using Genetic Algorithm to generate a dispatching queue.
Job Submission
4
The dispatcher computing resources information
5
Determine the job waiting time and machine availability.
Result
8
6
Allocate to the Available machine
7
Select job

Figure 2: Workflow Diagram of the Proposed System

Figure 2 describes the workflow diagram of the proposed system. When jobs are submitted to the system, the scheduler stores the jobs directly on the cloud. The job scheduler periodically pulls jobs from the storage and performs some pre-processing activities like the conversion due date and processing time to binary numbers. The job scheduler uses genetic model for job sequencing, and the sequenced jobs are placed in the dispatching queue. The job dispatcher collects the jobs from the dispatching queue and determines the jobs waiting time and machine availability using the fuzzy model. The system uses the determined waiting and machine availability information to dispatch jobs to the resources. The dispatcher also computes the resources information to know the available resources before transferring jobs to the available device for processing. The identical parallel machines are used to execute the allocated jobs. The dispatcher will first ascertain the availability of the device before allocating jobs to it.

A. Detailed Analysis of the Proposed System.

The job scheduler is one of the major components in this proposed system architecture. The activities of the job scheduler include the periodical pull of job from the storage, the job discovery, the job pre-processing and the job sequencing.

The periodical pull of job from the storage

The periodical pull of job from the storage is one of the activities of the job scheduler. It pools a set of jobs $N = \{J1, J2, \ldots, Jn\}$ of n jobs to be scheduled on a set $M = \{M1, M2, \ldots, Mm\}$ of m parallel machines. The first objective is to find the optimal schedule $S = \{S1, S2, \ldots, Sm\}$ where S_j is a subset of jobs assigned to machine Mj such that $max\{C1(S), C2(S), \ldots, Cm(S)\} = Cmax(S)$ is minimum where,

$$C_j(s) = \sum_{Pi \sqsubset Sj} pi \qquad (1)$$

The Job Discovery

The job scheduler starts to discover the attributes of the individual jobs that have been pulled from the storage. The index of machine is represented with I, $I = 1,2,\ldots\ldots,m$ where m is number of machines. $\{S1, S2, \ldots S_j, \ldots, ki\ S\}$ represents sequence of jobs which are assigned to machine i. The job index is represented by S_j, where $j = 1,2,\ldots, ki$, ki represents number of total jobs to be processed by machine i;
$S\,j\,p$ ……. processing time of job S_j;
$S\,j\,ST$ …… Starting time of job S_j;
$S\,j\,c$ …… Completion time of job S_j;
$S\,j\,d$ …… Due date of job S_j;

The Identical Parallel Machine total tardiness can be represented mathematically in the following manner:

$$ST_{Si}= 0 \qquad (2)$$

Constraint (2) which describes starting time of the machine is equal to zero.

$$C_{Si} = ST_{Si} + P_{Si} \qquad (3)$$

Constraint (3) denotes the completion time of the job in the sequence which is the summation of starting time and the processing time of the particular job in the sequence.

$$STs_j = Cs_{j-1}, \quad j > 1 \qquad (4)$$

$$C_{Sj} = ST_{Sj} + P_{Sj} \qquad (5)$$

The Job Pre-Processing

The job pre-processing does the binary conversion of all the data that have been discovered by the job scheduler.

The Job Sequencing

The job scheduler uses the pre-processed data for job sequencing. It makes use of genetic model approach, which consists of defining the population, estimating the chromosome the use of health feature and decision of parent chromosomes to operate genetic operator on them in order to generate new children chromosome.

Population Initialization

The chromosome shape is described as a mixture of two strings, SQ and SM of length same as the wide variety of jobs. Scheduling Queue (SQ) continues superiority constraints between jobs, and an object in TS denotes a job to be scheduled. An entry in Scheduling Machine (SM) represents the parallel machines the corresponding challenge is scheduled onto.

Estimation and Selection

The chromosomes are evaluated the usage of the fitness function. The health characteristic is defined as:

$$F (x) = (1/x) \qquad (6)$$

where: x = S j p ……. processing time of job Sj;

After evaluating the health values of all the chromosomes the higher health value chromosome are selected.
Crossover
Crossover mechanism reproduces new teens chromosomes which have some parts of both parents' chromosomes. The kind of crossover that was once used is a single-point crossover. Bits between successive crossover points are exchanged, producing two new offspring. In this randomly pick one or the second phase and apply two extraordinary crossover operators for these two parts. two Details about crossover are given in following steps:
C1: Input the Crossover probability Pc.
C2: Randomly choose pairs of chromosomes and generate a drift range (FLC) between 0 and 1 for every pair.
C3: If FLC &lt;= Pc, then recur step CR4 to step CR5, in any other case unswervingly reproduce those two chromosomes to the next generation.
C4: Arbitrarily produce two crossover points, p and q, between 1 and v and crossover flag CF between 0 and 1.
C5: If CF=0 then rearrange the order of duties in SQ between p and q of one chromosome according to the order of duties of some other chromosome, the rest of the two chromosomes are continued, in any other case trade the section in SP between p and q of two chromosomes and the rest of the two chromosomes are remained.

Crossover operation is carried out to produce new offspring (children). The crossover factor is particular and based on the crossover point; single point crossover is performed and a new offspring is produced. The regular genetic algorithm makes use of single factor crossover, the place the two mating chromosomes are cut once at corresponding factors and the sections after the cuts exchanged. Here, a cross-site or crossover point is selected randomly alongside the length of the mated strings and bits next to the cross-sites are exchanged. If terrific website is chosen, higher youth can be obtained by using combining top parents; otherwise it severely hampers string quality.

B. Determining machine failure and machine availability

The system takes the output of the job scheduler as an input to the job dispatcher and performs a fuzzy reasoning on it, considering the job waiting time and machine availability in order to know the jobs to be dispatch to the parallel machines. The fuzzification module will receive the crisp numeric values input, technique them and map them into fuzzy membership characteristic values. The fuzzy engine is responsible for processing all calculated membership characteristic values the usage of fuzzy sets' calculations and communicates with fuzzy rule base to identify the most suitable fuzzy output. However, the defuzzification module is responsible for changing the fuzzy output into a numeric output appropriate for the environment choice and manipulate situation. The fuzzy inference machine determines the membership characteristic for the job waiting time and machine availability which will be used in job allocation to the parallel machines.

Fuzzy Input/Output Specifications

The fuzzy input/output variables are used for the job dispatching Table 1: Fuzzy variables (I/O Specifications), assumed to be the base for the dispatching rule. The fuzzy I/O specification is in Table 1. There are two enter variables and one output. The output is the effect of the end result of entering states.

Fuzzification

The fuzzification is the procedure of remodeling a crisp cost into a fuzzy set, so that it can be used and processed via fuzzy inference mechanism. The inputs and output of the design as precise in Table 1 are assigned linguistic variables and some stages of membership.. For inputs one (the job waiting time) the corresponding range is short (0.0, 0.3), average (0.4, 0.6) and long (0.7, 1.0). For input two (machine availability) the corresponding value is failed = 0.0, many free = 0.25, two free = 0.5, one free = 0.75 and all busy = 1.0. The levels for output (job allocation) are retrieving job and send to next available device = 0.0, wait = 0.5, and allocate = 1.0. The fuzzy variables for this proposed model includes the following

Machine availability (MA) = {One Free (OF), Two Free (TF), Many Free (MF), All Busy (AB), Failed (F)}
Waiting Time (WT) = {Short (S), Average (A), Long (L)}
Job Allocation (JA) = {Allocate (A), Wait (W), Retrieve and Send to Available Machine (RSAM)}

Table 1: Fuzzy variables (Input/Output Specifications)

INPUT		**OUTPUT**
Machine availability (MA)	**Waiting Time (WT)**	**Job Allocation (JA)**
One Free (OF) Two Free (TF) Many Free (MF) All Busy (AB) Failed (F)	Short (S) Average (A) Long (L)	Allocate (A) Wait (W) Retrieve and Send to Available Machine (RSAM)

Fuzzy Rules

The rules are formulated using a series of if-then statements, combined with AND/OR operators. For instance, if one device is free AND the job waiting time is small, THEN job allocation should wait. With two inputs having eight membership functions, we have 5 * 3 = 15 rules as showed in Table 2.

R1: If MA = OF, and WT = S then JA = W
R2: If MA = OF, and WT = A then JA = W
R3: If MA = OF, and WT = L then JA = A
R4: If MA = TF, and WT = S then JA = W
R5: If MA = TF, and WT = A then JA = A
R6: If MA = TF, and WT = L then JA = A
R7: If MA = MF, and WT = S then JA = A
R8: If MA = MF, and WT = A then JA = A
R9: If MA = MF, and WT = L then JA = A
R10: If MA = AB, and WT = S then JA = W
R11: If MA = AB, and WT = A then JA = W
R12: If MA = AB, and WT = L then JA = W
R13: If MA = F, and WT = S then JA = RSAM

R14: If MA = F, and WT = A then JA = RASM
R15: If MA = F, and WT = L then JA = RASM

Table 2: The Fuzzy Rule Base Table

No	Device Availability (DA)	Job Waiting Time (JWT)	Job Allocation (JA)
1	One Free (OF)	Short (S)	Wait (W)
2	One Free (OF)	Average (A)	Wait (W)
3	One Free (OF)	Long (L)	Allocate (A)
4	Two Free (OF)	Short (S)	Wait (W)
5	Two Free (OF)	Average (A)	Allocate (A)
6	Two Free (OF)	Long (L)	Allocate (A)
7	Many Free (MF)	Short (S)	Allocate (A)
8	Many Free (MF)	Average (A)	Allocate (A)
9	Many Free (MF)	Long (L)	Allocate (A)
10	All Busy (AB)	Short (S)	Wait (W)
11	All Busy (AB)	Average (A)	Wait (W)
12	All Busy (AB)	Long (L)	Wait (W)
13	Failed (F)	Short (S)	Retrieve and Send to Available Machine (RSAM)
14	Failed (F)	Average (A)	Retrieve and Send to Available Machine (RSAM)
15	Failed (F)	Long (L)	Retrieve and Send to Available Machine (RSAM)

Defuzzification

The transformation from a fuzzy set to a crisp value is called defuzzification. The CoG is adopted in this study for defuzzification because its computational complexity is relatively high. It is depicted in Equation 7.

$$COG(B^0) = \frac{\sum_{q=1}^{N_q} \mu_{B^0}(y_q) y_q}{\sum_{q=1}^{N_q} \mu_{B^0}(y_q)} \qquad (7)$$

Where Nq is the number of quantization used to discretize membership function $\mu_B{}^0(y)$ of the fuzzy output B^0. $\mu_B{}^0(y)$ is the degree of membership and y_q are elements of the set.

Crisp Output = {Sum (Membership Degree * Singleton Position)}/(Membership degree) for instance, with the output membership degree, are retrieving job and send to next available device = 0.0, wait = 0.5, allocate = 1.0, then the crisp value will be **Crisp Output =** (0.1*0.00)+ (0.5*0.50)+(1.0*1.00)/(0.0+0.5 +0.1) = 0.83, for this result, therefore, system allocates job.

IV Experiment and Result

The new system was implemented to automatically generate the schedule for all the jobs uploaded in the system. This system generates values from the jobs uploaded to it and the value includes the job processing time and job waiting time. The generated jobs for testing this new model have the following characteristics:

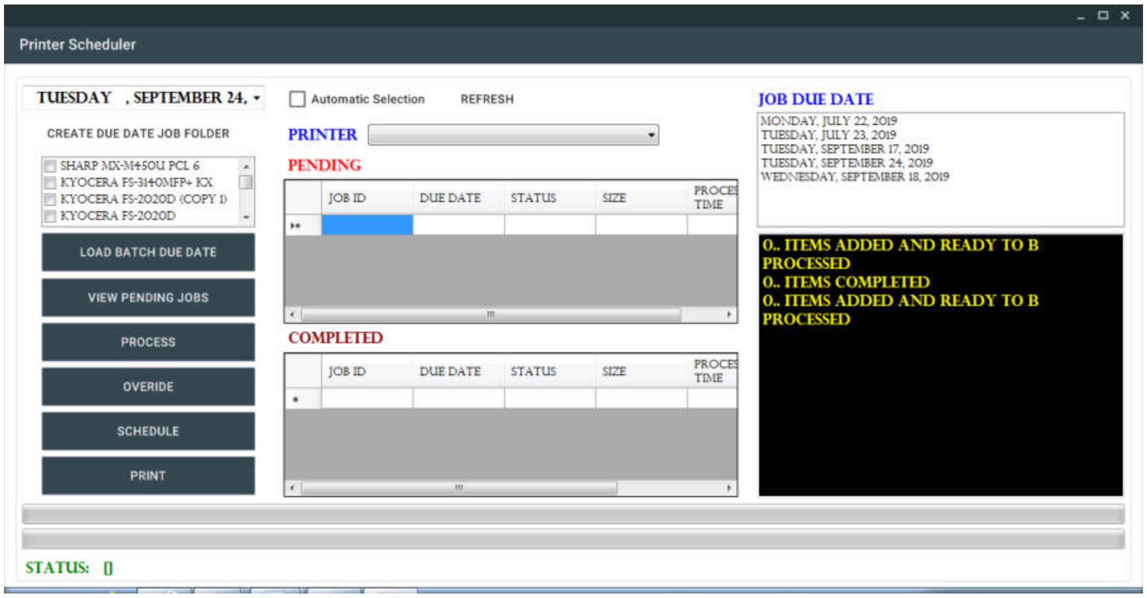
a. Size of uploaded job ranges from 200 to 1200 jobs with an interval of 100 jobs.

b. For the new model only one job enters the system at a time.
c. The processing time for each job is a random number between 1 mins and 4mins.
d. Number of machines used was fifteen for all the sizes of problems.

The developed model was used in the parallel machine environment. The result generated from the proposed system was compare with the result of other job scheduling algorithm like First Come First Sever (FCFS) scheduling model and genetic scheduling model. Three parameters were considered to evaluate the performance of the three different scheduling approaches. These parameters include the completion time, load balancing, and resource utilization.

Figure 3 shows the printer scheduler console for the new job scheduling model; from the console, jobs are uploaded to the parallel machines considering their due dates. The process automatically sequences jobs using the new model. The pending jobs and the completed job can be viewed from this control interface.

Figure 4 shows the job scheduling controls interface with the uploaded jobs displaying the job ID, the job processing time , the job size, job status and the scheduled due date. From the Figure, we can see the number of jobs uploaded in the system, the entire job is still waiting for the system to do the sequencing and no job has been completed. The process automatically sequenced the jobs based on the new model. The pending jobs and the completed job are viewed from this control interface.



TUESDAY , SEPTEMBER 24,

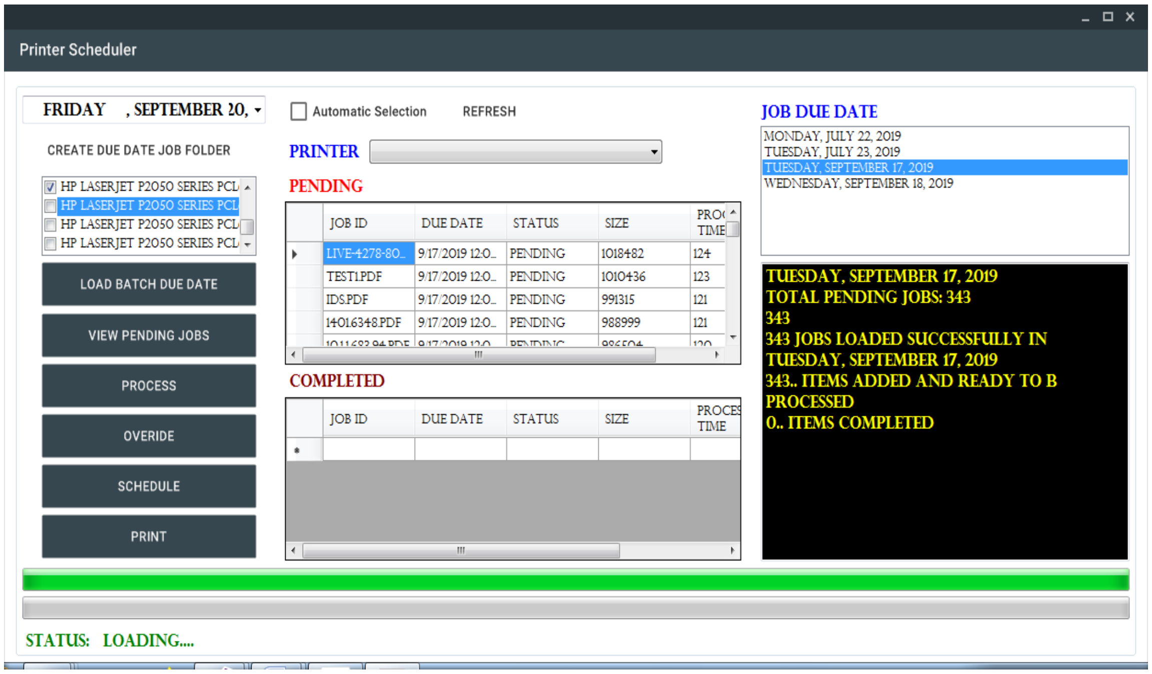
SHARP MX-M450U PCL 6
KYOCERA FS-3140MFP+ KX
KYOCERA FS-2020D (COPY 1)
KYOCERA FS-2020D

MONDAY, JULY 22, 2019
TUESDAY, JULY 23, 2019
TUESDAY, SEPTEMBER 17, 2019
TUESDAY, SEPTEMBER 24, 2019
WEDNESDAY, SEPTEMBER 18, 2019

0.. ITEMS ADDED AND READY TO B PROCESSED
0.. ITEMS COMPLETED
0.. ITEMS ADDED AND READY TO B PROCESSED

STATUS: []

Figure 3: Printer Scheduler Console for the new job scheduling model.


Printer Scheduler

FRIDAY , SEPTEMBER 20,

Automatic Selection REFRESH

CREATE DUE DATE JOB FOLDER

[✓] HP LASERJET P2050 SERIES PCL
HP LASERJET P2050 SERIES PCL
HP LASERJET P2050 SERIES PCL
HP LASERJET P2050 SERIES PCL

LOAD BATCH DUE DATE
VIEW PENDING JOBS
PROCESS
OVERIDE
SCHEDULE
PRINT

PRINTER

PENDING

JOB ID	DUE DATE	STATUS	SIZE	PROC TIME
LIVE-4278-80...	9/17/2019 12:0...	PENDING	1018482	124
TEST1.PDF	9/17/2019 12:0...	PENDING	1010436	123
IDS.PDF	9/17/2019 12:0...	PENDING	991315	121
14016348.PDF	9/17/2019 12:0...	PENDING	988999	121
1011683.94.PDF	9/17/2019 12:0...	PENDING	986504	120

COMPLETED

JOB ID	DUE DATE	STATUS	SIZE	PROCES TIME

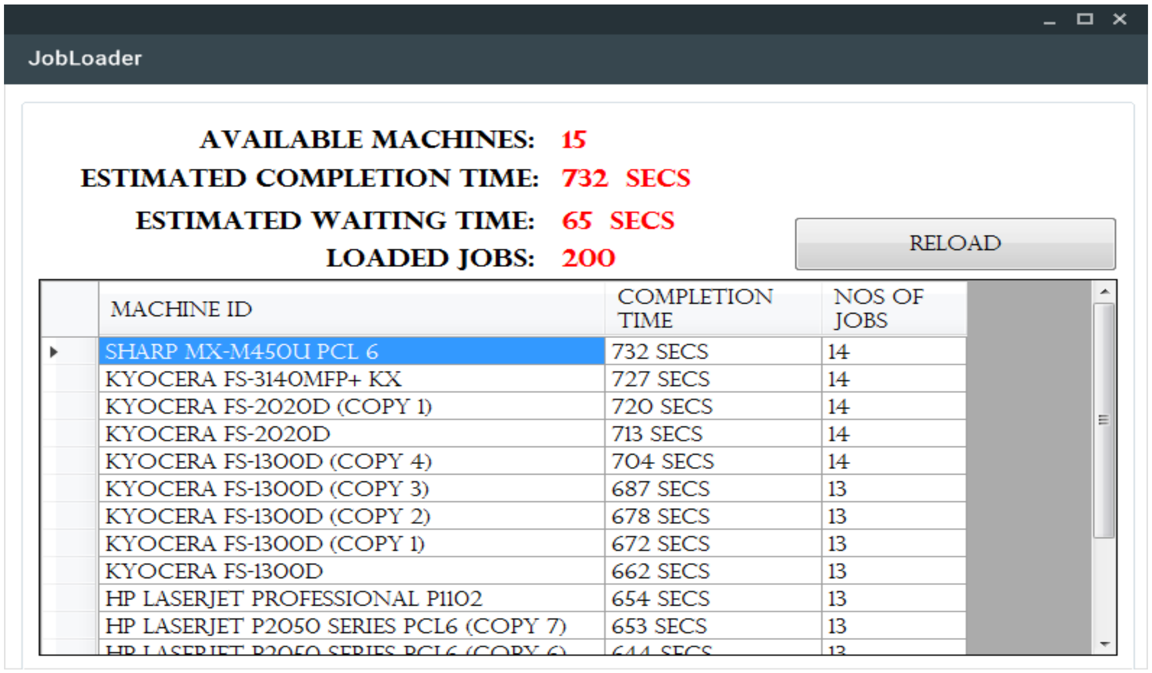
JOB DUE DATE

MONDAY, JULY 22, 2019
TUESDAY, JULY 23, 2019
TUESDAY, SEPTEMBER 17, 2019
WEDNESDAY, SEPTEMBER 18, 2019

TUESDAY, SEPTEMBER 17, 2019
TOTAL PENDING JOBS: 343
343
343 JOBS LOADED SUCCESSFULLY IN
TUESDAY, SEPTEMBER 17, 2019
343.. ITEMS ADDED AND READY TO B
PROCESSED
0.. ITEMS COMPLETED

STATUS: LOADING....

Figure 4: Printer scheduler console for the new job scheduling model displaying the pending jobs.

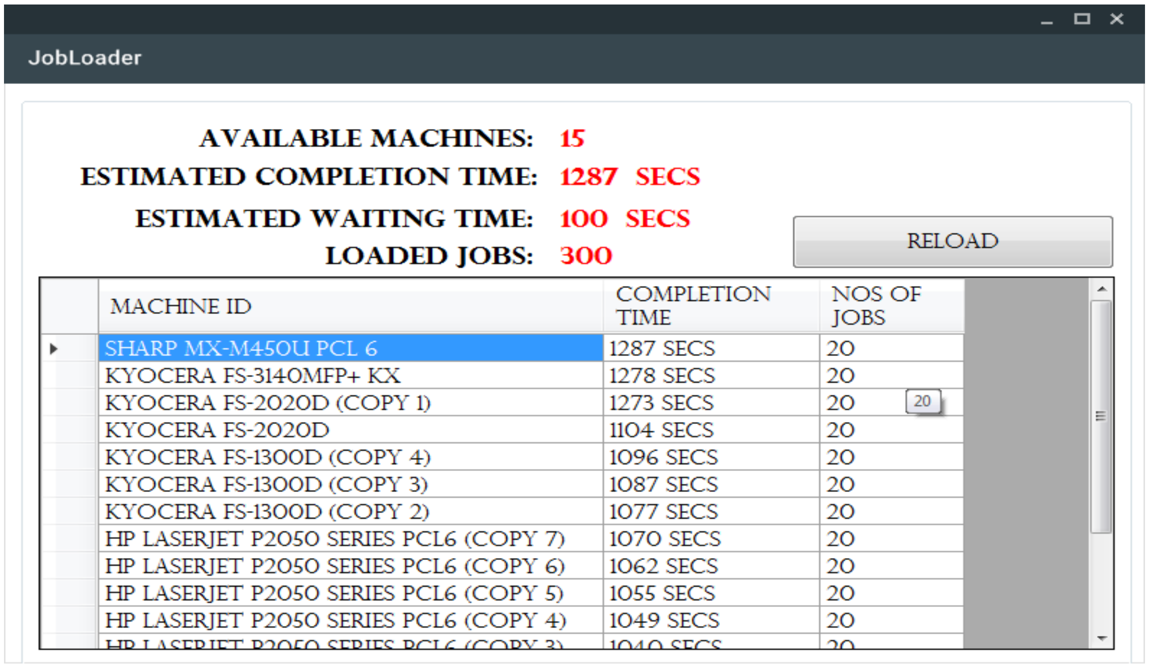

JobLoader

AVAILABLE MACHINES: 15
ESTIMATED COMPLETION TIME: 732 SECS
ESTIMATED WAITING TIME: 65 SECS
LOADED JOBS: 200

RELOAD

MACHINE ID	COMPLETION TIME	NOS OF JOBS
SHARP MX-M450U PCL 6	732 SECS	14
KYOCERA FS-3140MFP+ KX	727 SECS	14
KYOCERA FS-2020D (COPY 1)	720 SECS	14
KYOCERA FS-2020D	713 SECS	14
KYOCERA FS-1300D (COPY 4)	704 SECS	14
KYOCERA FS-1300D (COPY 3)	687 SECS	13
KYOCERA FS-1300D (COPY 2)	678 SECS	13
KYOCERA FS-1300D (COPY 1)	672 SECS	13
KYOCERA FS-1300D	662 SECS	13
HP LASERJET PROFESSIONAL P1102	654 SECS	13
HP LASERJET P2050 SERIES PCL6 (COPY 7)	653 SECS	13
HP LASERJET P2050 SERIES PCL6 (COPY 6)	644 SECS	13

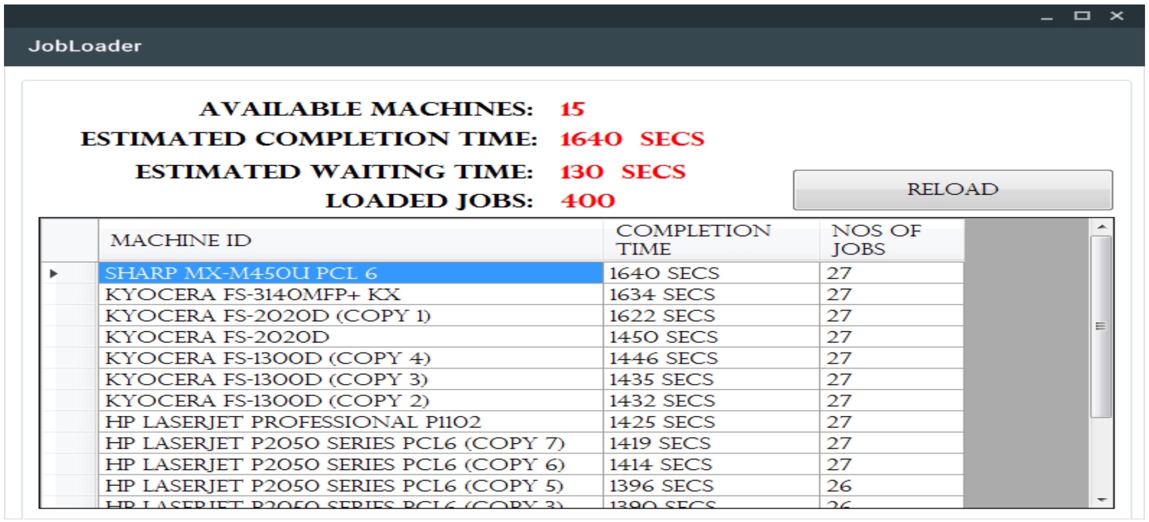
Figure 5: Estimated Completion Time of the parallel machines for 200 loaded jobs using the new scheduling mode.



AVAILABLE MACHINES: 15
ESTIMATED COMPLETION TIME: 1287 SECS
ESTIMATED WAITING TIME: 100 SECS
LOADED JOBS: 300

MACHINE ID	COMPLETION TIME	NOS OF JOBS
SHARP MX-M450U PCL 6	1287 SECS	20
KYOCERA FS-3140MFP+ KX	1278 SECS	20
KYOCERA FS-2020D (COPY 1)	1273 SECS	20
KYOCERA FS-2020D	1104 SECS	20
KYOCERA FS-1300D (COPY 4)	1096 SECS	20
KYOCERA FS-1300D (COPY 3)	1087 SECS	20
KYOCERA FS-1300D (COPY 2)	1077 SECS	20
HP LASERJET P2050 SERIES PCL6 (COPY 7)	1070 SECS	20
HP LASERJET P2050 SERIES PCL6 (COPY 6)	1062 SECS	20
HP LASERJET P2050 SERIES PCL6 (COPY 5)	1055 SECS	20
HP LASERJET P2050 SERIES PCL6 (COPY 4)	1049 SECS	20
HP LASERJET P2050 SERIES PCL6 (COPY 3)	1040 SECS	20

Figure 6: Estimated completion time of the parallel machines for 300 loaded jobs using the new job scheduling model.



AVAILABLE MACHINES: 15
ESTIMATED COMPLETION TIME: 1640 SECS
ESTIMATED WAITING TIME: 130 SECS
LOADED JOBS: 400

MACHINE ID	COMPLETION TIME	NOS OF JOBS
SHARP MX-M450U PCL 6	1640 SECS	27
KYOCERA FS-3140MFP+ KX	1634 SECS	27
KYOCERA FS-2020D (COPY 1)	1622 SECS	27
KYOCERA FS-2020D	1450 SECS	27
KYOCERA FS-1300D (COPY 4)	1446 SECS	27
KYOCERA FS-1300D (COPY 3)	1435 SECS	27
KYOCERA FS-1300D (COPY 2)	1432 SECS	27
HP LASERJET PROFESSIONAL P1102	1425 SECS	27
HP LASERJET P2050 SERIES PCL6 (COPY 7)	1419 SECS	27
HP LASERJET P2050 SERIES PCL6 (COPY 6)	1414 SECS	27
HP LASERJET P2050 SERIES PCL6 (COPY 5)	1396 SECS	26
HP LASERJET P2050 SERIES PCL6 (COPY 3)	1390 SECS	26

Figure 7: Estimated completion time of the parallel machines for 400 loaded jobs using the new job scheduling model.

Figure 5 shows the estimated completion time of the parallel machines for 200 loaded jobs using the new job scheduling model. From the Figure, we can view the available machines used for job processing, the estimated completion time for all jobs, the estimated waiting time and the number of job loaded. Figure 6 shows the estimated completion time of the parallel machines for 300 loaded jobs using the new job scheduling model. From the Figure, we can view the available machines used for job processing, the estimated completion time for all jobs, the estimated waiting time and the number of jobs loaded. Figure 7 shows the estimated completion time of the parallel machines for 400 loaded jobs using the new job scheduling model. From the Figure, we can view the available machines used for job processing, the estimated completion time for all jobs, the estimated waiting time and the number of jobs loaded.

The schedule length, average execution time, load balance and utilization can be computed from the generated result from the three tested scheduling models using the formula below:

1. $Schedule\ Length\ S_L = max\ \{E\_Time\ (\ )\}$
2. $Average\ Execution\ time\ Mi = \dfrac{\sum_{i=1}^{no\ of\ machines} \text{Execution time}}{\text{Total no of machines}}$
3. Load Balance = Average Execution time / Machine execution time
4. Utilization $= \dfrac{\sum_{i=1}^{no\ of\ machine} \left(\frac{\text{Machine Execution time}}{\text{machine Scheduling length}}\right)}{\text{Total no of machines}}$

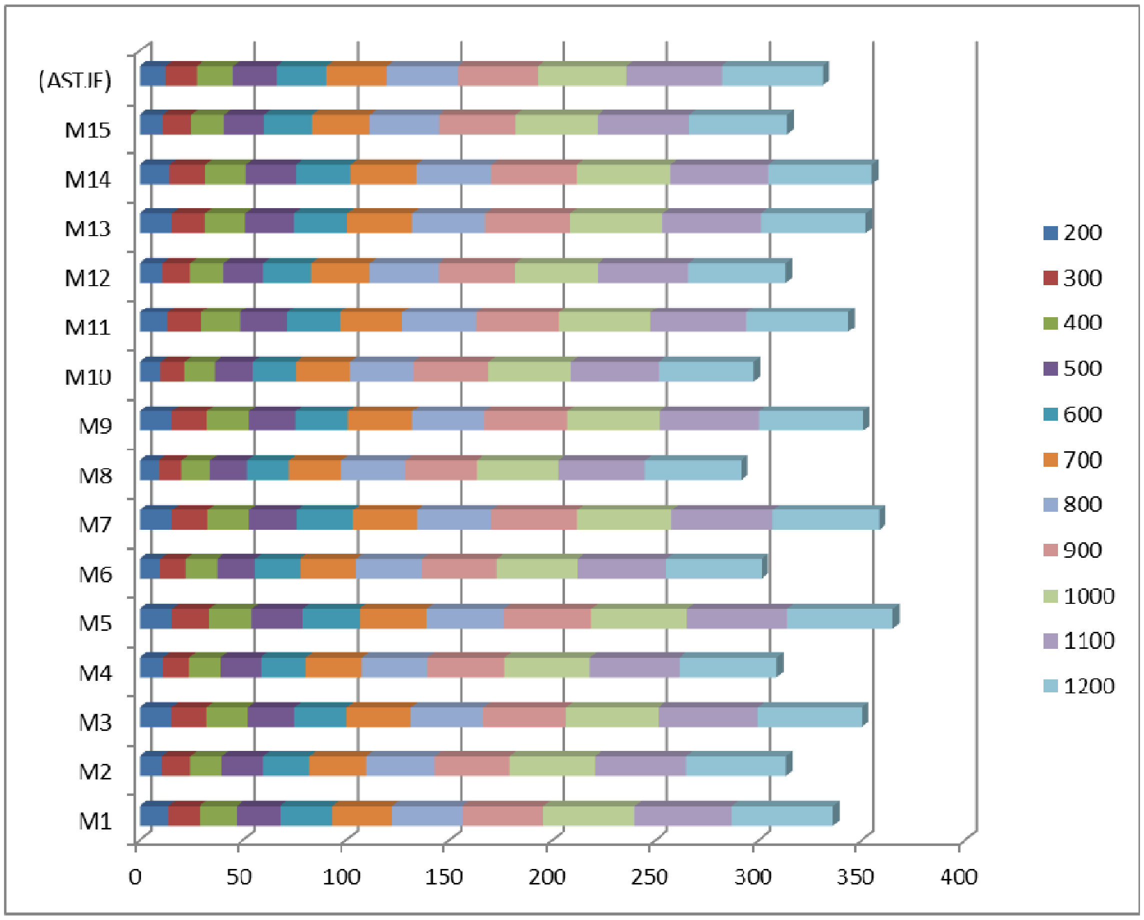


Figure 8. Graphical display of the performance analysis of scheduled time of job execution on individual machine using first come first serve (FCFS) scheduling approach.

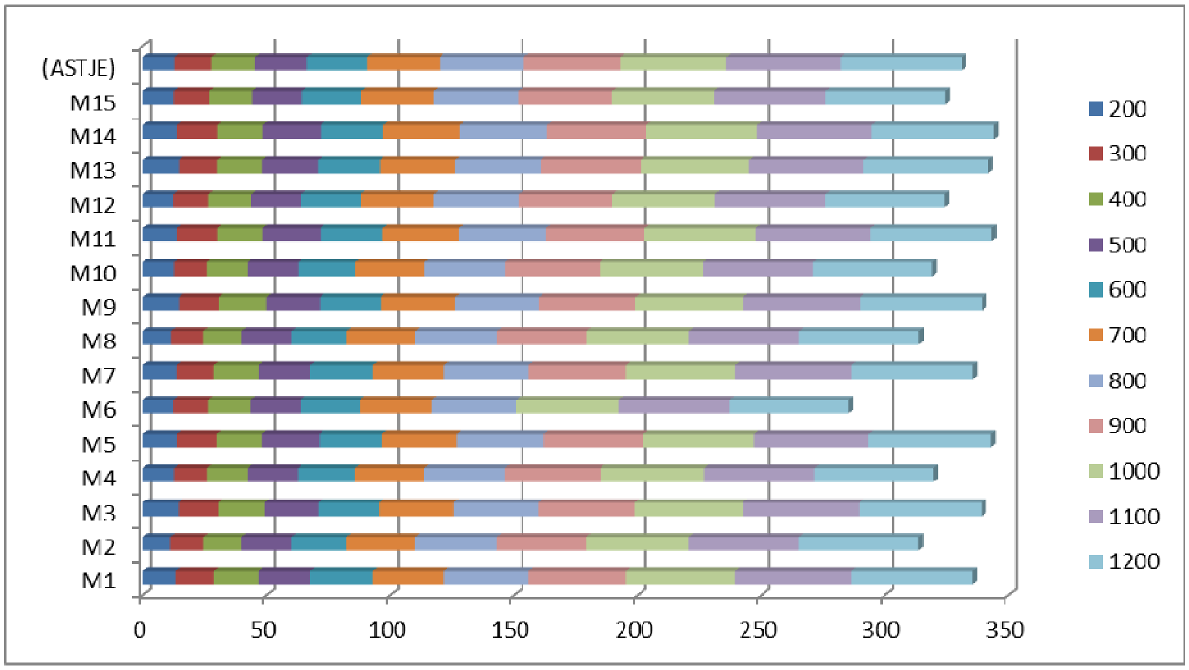


Figure 9. Graphical display of the performance analysis of scheduled time of job execution on individual machine using Genetic Algorithm (GA) scheduling approach.

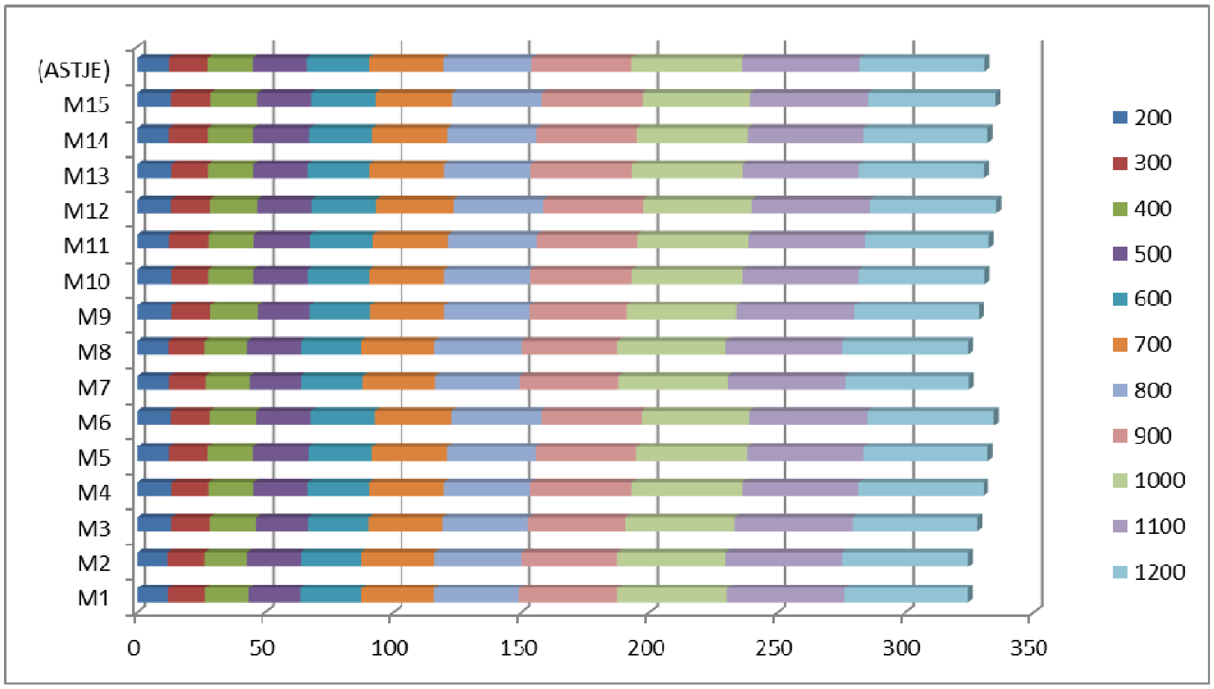


Figure 10. Graphical display of the performance analysis of scheduled time of job execution on individual machine using proposed model scheduling approach.

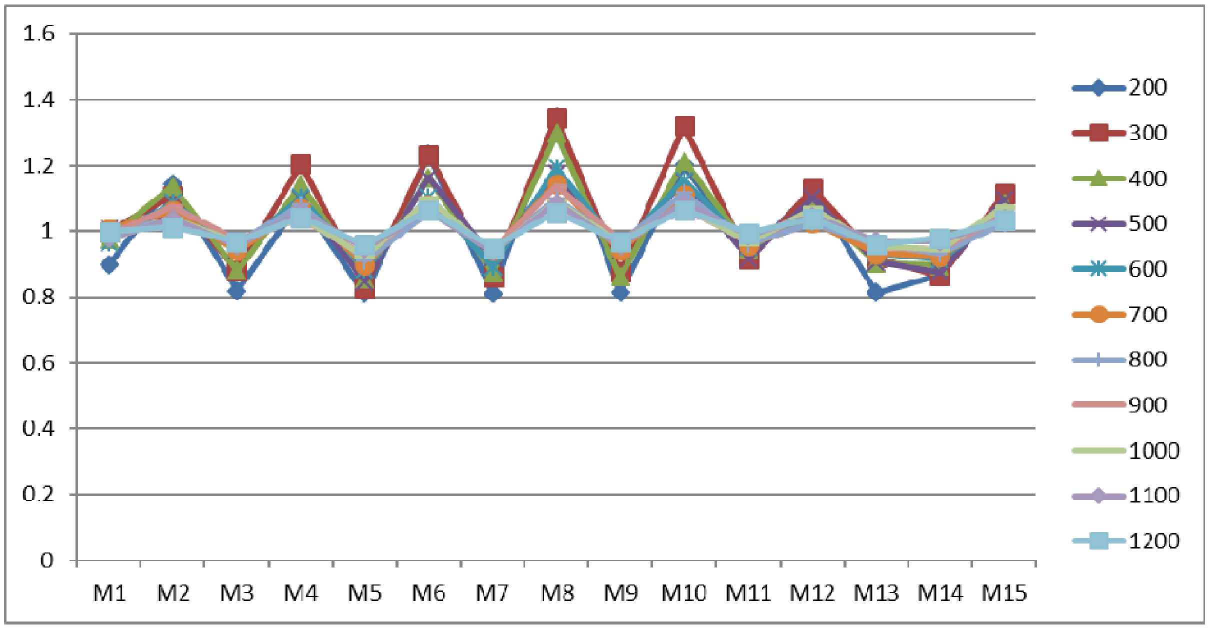


Figure 11: Graphical representation of load balancing across the individual parallel machine using FCFS job scheduling model.

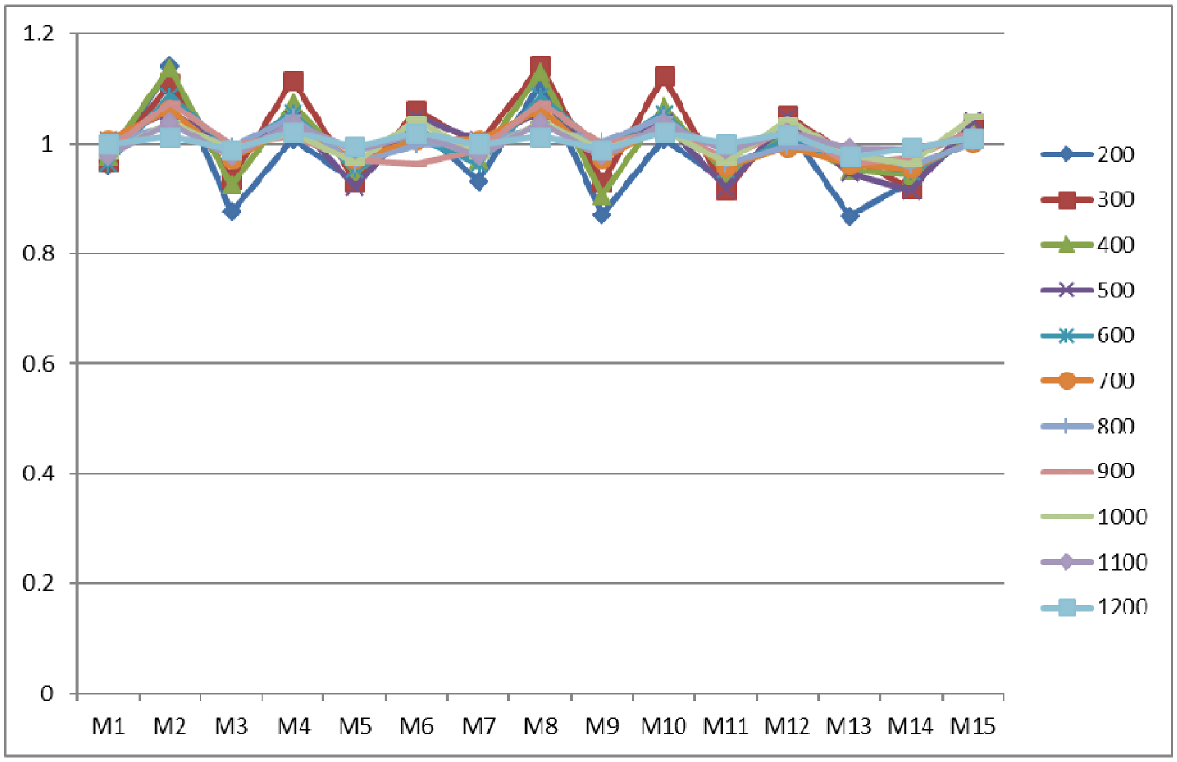


Figure 12: Graphical representation of load balancing across the individual parallel machine using GA job scheduling model.

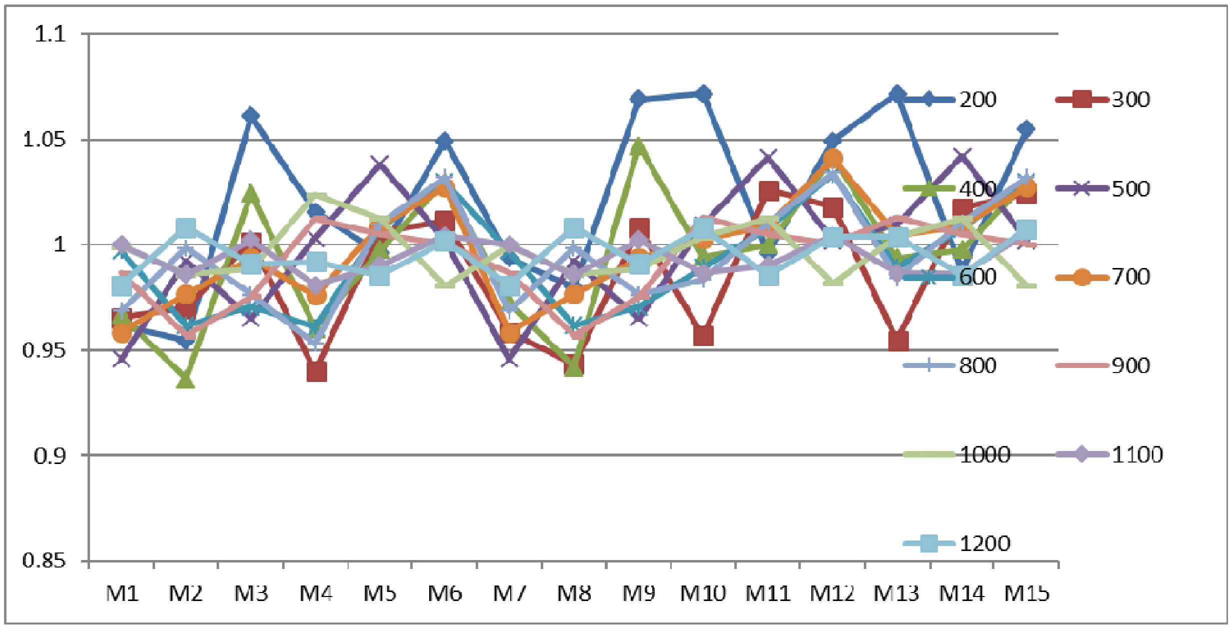


Figure 13. Graphical representation of load balancing across the individual parallel machine using the new job scheduling model.

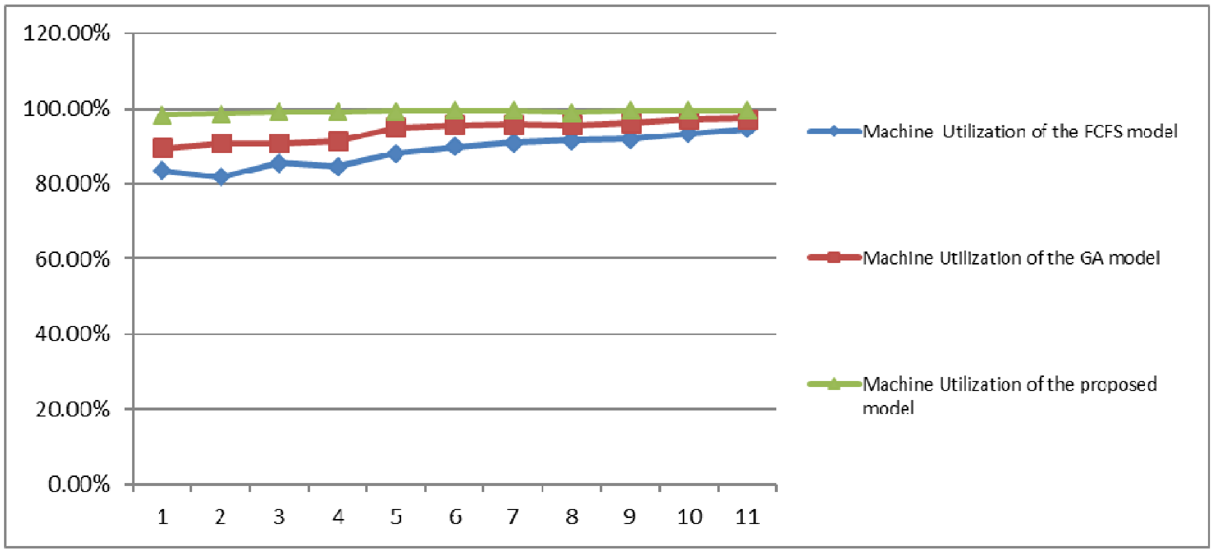


Figure 14: Graphical analysis of the machine utilization from the examined job scheduling algorithms.

V Result Discussion

Figure 8 shows graphical display of the performance analysis of scheduled time of job execution on individual machine using First Come First Serve (FCFS) scheduling approach. From the graph we can observe that there is big difference between the execution time of M5, M7, M9, M13, M14 and M8, M10, M6, M4, M12 respectively. The result shows that there is too much variation in the load balancing among the individual machines. Figure 9 shows graphical display of the performance analysis of scheduled time of job execution on individual machine using Genetic Algorithm (GA) scheduling approach. From the graph we can observe that there is big difference between the execution time of M5 and M8. The result shows that there is better load balancing among the individual machines when compared with the First Come First Serve (FCFS) scheduling approach. Figure 10 shows graphical display of the performance analysis of scheduled time of job execution on individual machine using new model scheduling approach. From the graph we can observe that there is small difference between the execution time of individual machine. The result shows that there is better load balancing among the individual machines when compared with the first come first serve (FCFS) scheduling approach and Genetic Algorithm (GA) scheduling approach.

Figure 11 shows graphical representation of load balancing across the individual parallel machine using FCFS job scheduling model. From the graph we can observe that the load balancing across the individual machine was from 0.8 to 1.4, which can be seen as a wide margin. The result does not show good load balancing across individual systems. Figure 12 shows graphical representation of load balancing across the individual parallel machines using GA job scheduling model. From the graph we can observe that the load balancing across the individual machine was from 0.8 to 1.2, which is better than FCFS job scheduling model. The result shows a better load balancing across individual systems when compared with FCFS job scheduling model. Figure 13 shows graphical representation of load balancing across the individual parallel machines using the new job scheduling model. From the graph we can observe that the load balancing across the individual machine was from 0.93 to 1.0, which is better than FCFS job scheduling model and GA job scheduling model. The result shows a better load balancing across individual systems when compared with FCFS job scheduling model and GA job scheduling model.

Figure 14 shows graphical analysis of the machine utilization from the examined job scheduling algorithms. From the graph, it can be observed that the new model achieved better resource utilization than the FCFS model and the GA models; this shows that the new system achieved high machine utilization when compared with other existing model.

VI Conclusion

This study has shown efficient job sequencing and dispatching model that decreases the total time for jobs execution in identical parallel machine system. It is an effective job scheduling model that takes care of the different issues to obtain good job schedule. The research was able to handle the issue of premature convergence, which is often encountered by genetic algorithm. The system also considers the due date of the individual jobs which is an advantage when compared with existing systems. The result generated from experiment conducted in the course of this research has shown that the system achieves load balancing among the individual machines and high machine utilization, thereby minimizing the total job execution time.

References

Abraham, R. B. and B. Nath. (2000), Nature's Heuristics for Scheduling Jobs on Computational Grids, The 8th IEEE International Conference on Advanced Computing and Communications (ADCOM 2000), 5(3), 45-52,.

Antonio J.S, J.Y Antonio, E.M Jose., G. G Sebastian and P. Rocio, (2011), A Multi-Criteria Meta-Fuzzy-Scheduler For Independent Tasks In Grid Computing, *Computing and Informatics, 30(2); 1201–1223.*

Braun R., H. Siegel., N. Beck, L. Boloni., M Maheswaran, A. Reuther, J. Robertson., M. Theys, B. Yao ,D Hensgen. and R. Freund.,(2001), "A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems", Journal of Parallel and Distributed Computing, 61: 810-837,

Fayçal B, S. Zaki, and S. Mehdi, (2013), A Genetic Algorithm for the Parallel Machine Scheduling Problem with Consumable Resources *International Journal of Applied Meta heuristic Computing, 4(2), 17-30,*

Fuqing Z, Z. Jianlin, Z. Chuck and W. Junbiao, (2015), An improved shuffled complex evolution algorithm with sequence mapping mechanism for job shop scheduling problems, *Expert Systems with Applications, 42(2); 3953–3966*

Jasbir S and S. Gurvinder (2012), Task Scheduling using Performance Effective Genetic Algorithm for Parallel Heterogeneous System, *International Journal of Computer Science and Telecommunications 3(3); 233 - 245*

Jaspal S and S. Harsharanpal, (2011), Efficient Tasks scheduling for heterogeneous multiprocessor using Genetic algorithm with Node duplication, *Indian Journal of Computer Science and Engineering (IJCSE), ISSN : 0976-*

5166, 2(3); 112 - 123

Kamaljit K, C. Amit and S. Gurvinder (2010), Heuristics Based Genetic Algorithm for Scheduling Static Tasks in Homogeneous Parallel System,*International Journal of Computer Science and Security (IJCSS), 4 (2); 183-198*

Karthick K. U. (2011), "A Dynamic Load Balancing Algorithm in Computational Grid Using Fair Scheduling", IJCSI International Journal of Computer Science Issues, 8(5), 1- 11, .

Liang S, C. Xiaochun and L Yanchun, (2010), Solving Job Shop Scheduling Problem Using Genetic Algorithm with Penalty Function, *International Journal of Intelligent Information Processing ISSN 2093-1964, 1(2); 65-77.*

Lei Z., C. Yuehui, S. Runyuan, J. Shan and Y. Bo, (2006) "A Task Scheduling Algorithm Based on PSO for Grid Computing", IEEE, 2 (1), 26-34.

Marish K and Amit D, (2015), Reliable And Efficient Task Scheduling Algorithm Based On Genetic Algorithm In Cloud Computing Environment, *International Journal of Computing and Corporate Research, 5(6); 57 - 66 .*

Moghaddam T.R., B Javadi, F Jolai and A Ghodratnama, (2010), 'The use of a fuzzy multi-objective linear programming for solving a multi-objective single-machine scheduling problem', *Applied Soft Computing, 10(1), 919-925.*

Mostafa R. M and H. A. A. Medhat (2011), Hybrid Algorithm for Multiprocessor Task Scheduling, *IJCSI International Journal of Computer Science Issues, 8(3), 14- 23,* Ramamritham, K. and J. A. Stankovic, (1994), Scheduling Algorithms and Operating Systems Support for Real-time Systems, Proceedings of IEEE, 82(1), 55-67.

Neelu S and S Sampada (2012), Task Scheduling Using Compact Genetic Algorithm for Heterogeneous System, *International Journal of Advanced Research in Computer Engineering & Technology, ISSN: 2278 – 1323, 1(3); 218- 221*

Orosz J.E and S. H. Jacobson. (2002) , Analysis of static simulated annealing algorithm, Journal of Optimization theory and Applications, 4(2), 165-182,.

Rachhpal S (2012),Task Scheduling With Genetic Approach and Task Duplication Technique, *International Journal of Computer Applications & Information Technology 1(1); 1-11*

Rachhpal S (2014),Task Scheduling in Parallel Systems using Genetic Algorithm, *International Journal of Computer Applications, 108(16); 0975 – 8887*

Rachhpal S (2016), An Optimized Task Duplication Based Schedulingin ParallelSystem, *I.J. Intelligent Systems and Applications,* 8, 26-37

Rajakumar S., V.P Arunachalam, and Selladurai V., (2006), Workflow balancing in parallel machine scheduling with precedence constraints using genetic algorithm, Journal of Manufacturing Technology Management, 17(2), 20-34.

Rajendra P, and V Rakesh, (2015), Adaptive job scheduling on computational grid for resource allocation and job completion Based on Feature Selection*, International journal of Master of Engineering Research and Technology 2(3); 77 - 86*

Ramkumar R., A. Tamilarasi and T. Devi, (2011). Multi Criteria Job Shop Schedule Using Fuzzy Logic Control for Multiple Machines Multiple Jobs, International Journal of Computer Theory and Engineering, 3(2), 282-286 [5]

Sachi G, A. Gaurav and K. Vikas, (2013) An Efficient and Robust Genetic Algorithm for Multiprocessor Task Scheduling, *International Journal of Computer Theory and Engineering, , 5(2), 377-382*

Safwat A. H and A. O Fatma (2016*)* Genetic-Based Task Scheduling Algorithm in Cloud Computing Environment, *(IJACSA) International Journal of Advanced Computer Science and Applications, 7 (4). 112 – 122*

Savas B(2011) Non-identical parallel machine scheduling using genetic algorithm, Expert Systems with Applications 38 (11) 6814–6821

Song S., Y. Kwok and K. Hwang, (2005). "Security-Driven Heuristics and A Fast Genetic Algorithm for Trusted Grid Job Scheduling", IEEE International Parallel and Distributed Processing, 4(2), 65-74.

Selvi. V (2014), Multi Objective Optimization Problems On Identical Parallel Machine Scheduling Using Genetic Algorithms, International Journal on Recent Researches in Science, Engineering & Technology, 2 (7) 112 - 122,

Thamilselvan R. and P Natesan. (2016), Hybridization of Genetic Algorithm with Cuckoo Search Algorithm for Job Scheduling on Distributed Heterogeneous Computing Systems, *Asian Journal of Research in Social Sciences and Humanities, ISSN 2249-7315. 6(9); 2162-2175*

Tufan D, O. Vildan, C. D Nihan and T. Belgin. (2011) A Genetic Algorithm Approach for Minimizing Total Tardiness in Parallel Machine Scheduling Problems, *Proceedings of the World Congress on Engineering, London, U.K.*2(1) 72 -86

Turker A. K. and C. Sel, (2011) Scheduling Two Parallel Machines with Sequence-dependent Setups and A Single Server Gazi University Journal of Science, 24(1);113-123

Weinberg J , (2002), "Job Scheduling on Parallel Systems", Job Scheduling Strategies for Parallel Processing, 5 (1), 67-73.

Xiaofeng X and G. U. Yuesheng (2010) Global Optimization Based on Hybrid Clonal Selection Genetic Algorithm for Task Scheduling, *Journal of Computational Information Systems 6(1) 253-261*

Ye L and C. Yan (2010), A Genetic Algorithm for Job-Shop Scheduling, *Journal of Software, 5(3); 212- 127*