ISSN:0975-9646

Analysis of Scheduling Algorithm for Ant Colony Optimization

Ayushi[1] and P. Abrol[2]
[1,2] Department of Computer science & IT,
University of Jammu, 180006, J&K, India

Abstract : Scheduling of different processes has a significant effect on the overall performance of the system. Efficient scheduling of jobs to incongruous processors for any application is difficult to achieve for high performance. This paper presents the comparative analysis of scheduling algorithms for n-processes based on ant colony optimization (ACO) algorithm. This study uses a prototype using ACO for the arrival of processes at a schedule. ACO is an affective metaheuristic optimization technique used for finding the optimal solution for single as well as multi-variant problems. Moreover, the results present the deterministic scheduling algorithms (First Come First Serve and Shortest Job First) for the comparative analysis in the tabulated format using ACO technique for average waiting time and average turnaround time.of the system. Efficient scheduling of jobs to incongruous processors for any application is difficult to achieve for high performance. This paper presents the comparative analysis of scheduling algorithms for n- processes based on ant colony optimization (ACO) algorithm. This study uses a prototype using ACO for the arrival of processes at a schedule. ACO is an affective metaheuristic optimization technique used for finding the optimal solution for single as well as multi-variant problems. Moreover, the results present the deterministic scheduling algorithms (First Come First Serve and Shortest Job First) for the comparative analysis in the tabulated format using ACO technique for average waiting time and average turnaround time.

Index Terms ACO, metaheuristic, FCFS, SJF, average waiting time, average turnaround time, scheduling, processes.

I. Introduction

The incongruous computing platform meets the computational demands of various problems. In this type of platform, different types of jobs are executed in a sequence. The main key challenge of such incongruous platforms is effective scheduling [1]. Scheduling is one of the major factors in operating system (OS) that affects the overall performance of the system. By scheduling the processes and assigning different jobs to the processors in a specified sequence to maximize the systems efficiency. When more than one process are ready to execute in the ready queue, then the module of OS decides to use the schedulers(scheduling algorithms) that decides which process will be executed first. Scheduling of processes basically is the mapping of set of jobs to the set of processors which avoids the situation where some of the processors are overloaded while others are idle [2]. Fig. (1) shows the diagrammatical structure where scheduling fits in the multi-variant processes and their requests in a very simplified manner [3].

There are different types of scheduling algorithms which satisfies the scheduling criteria of OS for maximizing CPU utilization. Various scheduling criteria's are: average turnaround time, average waiting time, average response time, burst time, etc. Also there are various scheduling algorithms each having its own characteristics, which are used for scheduling jobs in OS. They are: first come first serve algorithm, shortest job first algorithm, priority based scheduling algorithm, round robin algorithm, etc. [4]. Different types of scheduling processes are performed to solve different scheduling problems in order to improve their efficiency as shown in Fig. (1).

Several heuristic based algorithms are proposed to solve different types of scheduling problems. But we are using ant colony optimization (ACO) algorithm in our case to optimize the scheduling of processes. ACO finds the near optimal solution within the reasonable computational time.

II. Related Work

Several metaheuristic algorithms are proposed in case of scheduling problem. In (Braun *et al*., 2001) [5], the author has compared eleven metaheuristic algorithms for mapping and then scheduling a set of processes for minimizing the makespan of different processes. Many more different types of algorithms are defined such as: list-based scheduling algorithm (Radulescu and Gemund, 2002) [6], cluster-based and duplication-based scheduling algorithm (Ucar et al., 2006) [7]. In (Blum and Roli, 2003) [8], the author describes the one of the popular approximation optimization technique, i.e, ACO. This technique is inspired by the foraging behavior of real ants. In the further addition, (Blum, 2005) [9], author suggested that the chemical pheromone which is deposited by the ants on their trail path form food source to their colony is the core behavior of their indirect communication. According to Adhokshai mishra *et al* [10], Genetic algorithm is mostly used for process scheduling in different OS. Its main limitation is that there is no absolute assurance that it will ever find the global optimum solution. That's why ACO is best to be used for process scheduling as it gives better results for approximation and combinational problems.

III. Basic Aco Algorithm

Ant colony optimization (ACO) works as an optimization technique that was first introduced in early 1990's by Dorigo and Gamberdella. It is a probabilistic algorithm that uses pseudorandom proportional rule. It is a population based metaheuristic that tackles NP-hard discrete and combinational optimization problems [11] motivated by the sharp and foraging behavior of real ants, and in particular, how ants can find the shortest path between their colonies and the food source. Their indirect communication (a terminology known as Stigmergy) [12] is inspired by means of the trails of some chemical matter called pheromone. The role of pheromone is to guide the other ants to reach to the particular destination point. In Fig. 2, it is clearly observed that the quickest trail time on the shortest path makes the pheromone value high which is placed on that path. [13]

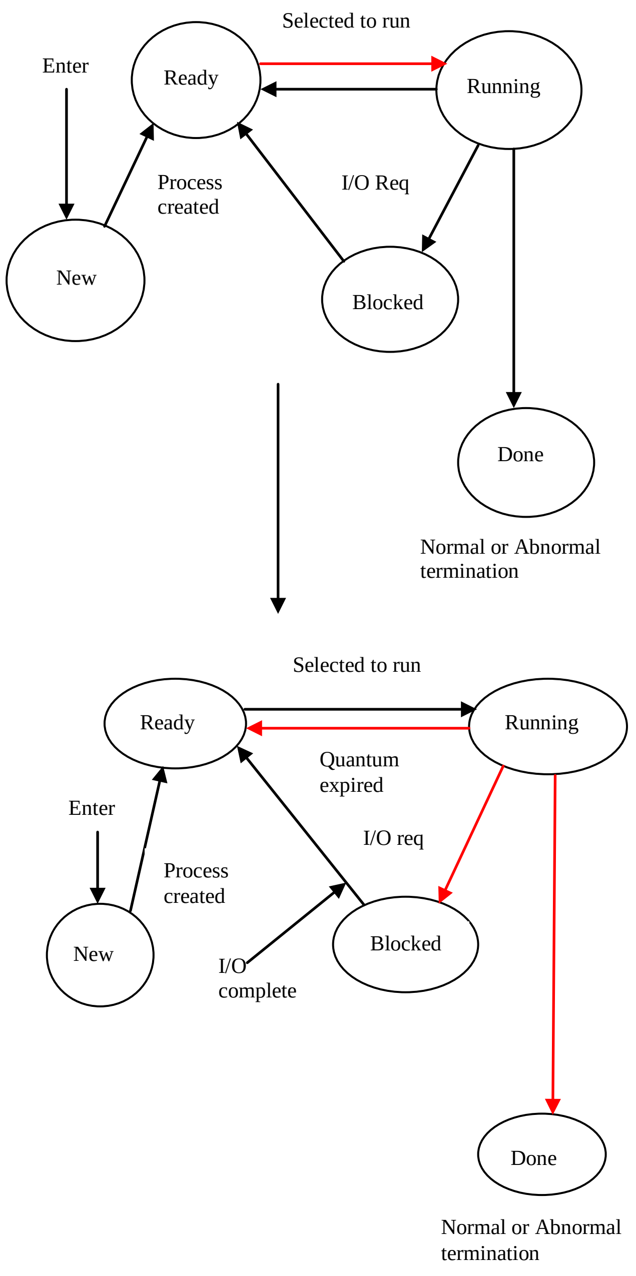

FIGURE 1. Scheduling Processes

From Fig. 2, it is clear that ant's navigate from nest (N) to the food source (F). As ants are blind, the shortest path is discovered through the higher concentration of pheromone value. Ant's move at random and deposits pheromone on their path. More pheromone on the path increases the probability of the path being followed.

In this paper, we are going to define a new type of general-purpose metaheuristic algorithm that can be used to solve different types of discrete-combinatorial optimization problems. The new defined metaheuristic algorithm has following desirable characteristics:

- *Versatile*: It can be applied to the same problem with similar versions. For example, asymmetric travelling salesperson problem is the straight forward extension of simple travelling salesperson problem.
- *Robust*: It can be easily applied to other combinatorial optimization problems with only minimal changes to them. For example, assignment problem and scheduling problem.
- *Population based approach*: It allows the exploitation of positive feedback as a new updated search mechanism.

These above mentioned desirable properties are counter balanced by the fact that for different applications, the ACO can be out performed by more specialized algorithms.

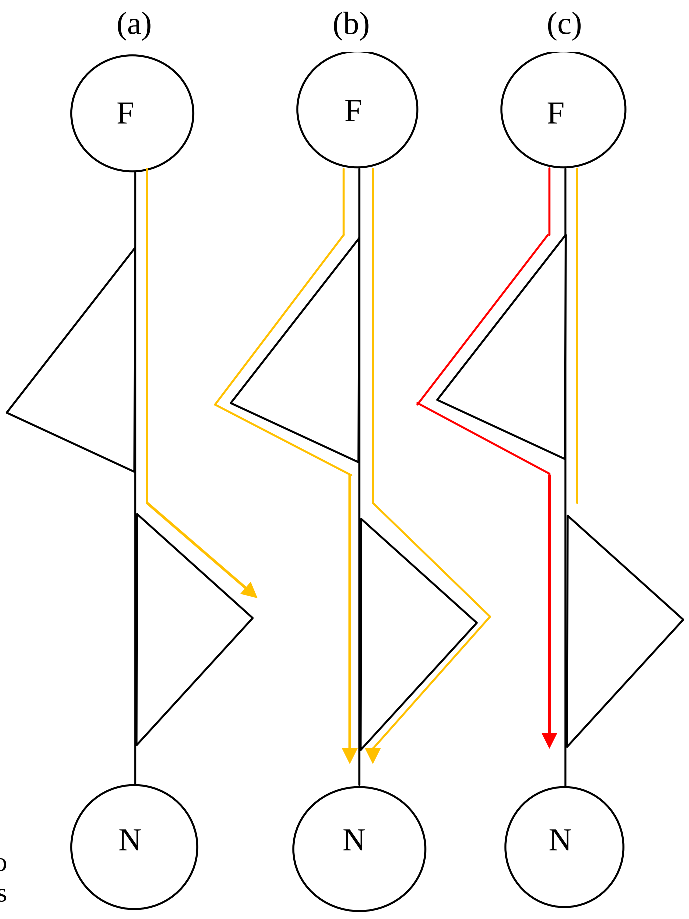

FIGURE 2 Ants navigation from nest (N) to food source (F) and vice-versa.

IV. Problem Statement

Efficient and strategic scheduling of jobs in incongruous processors for any type of application is difficult in order to achieve high performance. The core work is to find the feasible schedule for a given set of processors without exceeding their capacity, is actually, NP-hard. The study of this paradigm uses ACO for arriving at a schedule.

To solve the scheduling problem, there are m processors (p_1, p_2, p_3……. P_m) and all these processors have their static burst time. [14] Let there be n number of jobs with job set (j_1, j_2, j_3, ….., j_n). Therefore we will be having a utilization matrix U of size n*m where n be the number of jobs and m be the number of processors. Our main goal is to find the optimum sequence of processes in which the processes can be scheduled in such a method that total turnaround time and total waiting time should be minimum. The sample utilization matrix is shown in Table 1:

TABLE I
Utilization matrix with 3 processors and 3 jobs

Jobs	**Processes**		
	P_1	P_2	P_3
J_1	$U_{1,1}$	$U_{1,2}$	$U_{1,3}$
J_2	$U_{2,1}$	$U_{2,2}$	$U_{2,3}$
J_3	$U_{3,1}$	$U_{3,2}$	$U_{3,3}$

Where, $U_{i,j} \alpha P_i * J_j$

In the sample utilization matrix shown in above Table, the number of rows is equal to the number of jobs and number of columns is equal to the number of processes.

A. Assumptions made by CPU Scheduling Problem.

- There are m-processes waiting for the CPU allocation.
- All the processes are independent of each other and all are in a competition for the allocation of CPU.
- The purpose of scheduling algorithm is to allocate CPU to every process in some particular order so that neither process have to wait for CPU allocation nor CPU have to remain idle in order to optimize the performance criteria of the algorithm i.e, minimum turnaround time and minimum waiting time.
- The ACO based scheduling algorithm should be non-primitive i.e, no one can have the permission to get the CPU from one process and allocate it to another process during execution.
- The arrival time of all the processes should be zero.

V. Applying Aco To Shortest Path Problem

For a directed graph *G= (V, E)* where V is the set of vertices (V_1, V_2,…., V_n) and E is the set of edges (E_1, E_2,…., E_n), assign the total cost $a_{i,j}$ to each of its edges, where (i, j) ϵ E (this cost is basically the length of the tour). For the resulting path (n_1, n_2,…., n_k), its cost can be expressed as [15]:

$$a_{i,j} = \sum^{k-1}_{i=1} a_{ni}\ n_{i+1} \qquad (1)$$

According to the definition, a path is called the shortest path if it has the shortest length among all the paths from starting node to the terminating node.

The shortest path problem finds its application in various areas such as telecommunication problem, routing problem, etc. For finding shortest path in ACO, it uses some parameters, as shown in Table 2:

TABLE II
Parameters of ACO finding shortest path

Parameters	**Values**
M	number of Ants
Α	the parameter that defines the influence of pheromones on the choice of the next vertex
β	parameters that defines the influence of remaining data on the choice of next vertex
P	parameters that defines the speed at which pheromone evaporates, where ρ = [0, 1]
τ_o	initial level of pheromone on edges
τ_{min}, τ_{max}	minimum and maximum acceptable of pheromones on edges
s	starting node
t	terminating node

The number of ant's *m* influences the accuracy of the solution obtained as the result of the procedure of the algorithm. The parameters α and β modifies the method for the selection and updation of the new node or vertex, which in turn increases the quality of the solution.

Algorithm 1: ACO for shortest path problem

```
Initialize (G, s) for the shortest path
Initial vertex, s= 0; C <= 0
for al l i ϵ V do
   for all j ϵ V do
      $\tau_{i,j} <= \tau_o$
      $v_i$_edges$_{i,j}$ <= false
set (i,j) be the edges not visited
    if $a_{i,j}$ > C then C<= $a_{i,j}$
   end
   $V_i$_nodes$_i$ <= false
   Set i node not to be visited
end
cost <= C (V-1)
time<= 0
for k: 1 to m do
    reset (G,k)   // erase data gathered by ant k
    set Run (k,0)  //set the counter of ants route k to 0
   set node (k,s)  // set current vertex of ant k
   set Visited (k,s) // set vertex as visited for ant k
    Add (list, time, k)  // add ant k to list with time 0
end
convergence<= 0
Cost_length<= +∞
```

The parameters α and β modifies the method for the selection and updation of the new node or vertex, which in turn increases the quality of the solution. The parameter ρ defines the speed of the evaporation of the pheromone.

Faster the pheromone evaporates, less will be that path preferred for the next trail. Depending upon the whole procedure s and t are updated and final cost of the path is preferred. The above algorithm initiates the whole procedure and also uses all the above mentioned parameters [16].

As discussed in algorithm (1), we can have the trail updated intensity. Let $\tau_{i,j}$ (t) be the intensity of trail on edges (i,j) at time t. Let n be the total number of iterations of the algorithm for the completion of the tour. At this stage, the trail intensity is updated as given in the formula below [17]:

$$\tau_{i,j}(t+n) = \rho * \tau_{i,j}(t) + \Delta\, \tau_{i,j} \qquad (2)$$

Where, ρ is the coefficient such that (1 - ρ) is the evaporation rate between time t and (t + n) :

$$\Delta\, \tau_{i,j} = \sum{}^{m}{}_{k=1}\, \Delta\tau_{i,j} \qquad (3)$$

Where $\Delta\, \tau_{i,j}$ is the quantity of pheromone per unit length on edges (i, j). Now the transition probability by which ant k moves from starting point to destination is given below as:

$$P_{i,j}{}^{k}(t) = [\tau_{i,j}(t)]^{\alpha}.[n_{i,j}]^{\beta} / \sum{}_{k}{}^{allowed}\, [\tau_{i,j}(t)]\alpha \,.\, [n_{i,j}]_{\beta}$$
$$0\,; \qquad otherwise \qquad (4)$$

Where α and β are the parameters that controls the relative importance of trail path. The whole procedure of finding the shortest path in ACO depends upon the probability of moving of ants from one node to another and thus choosing the optimal one.

VI. Applying Aco To Cpu Scheduling Problem

ACO is a metaheuristic technique which is based on the smart behavior of ants. Given a set of incongruous multi-processors and job scheduling, the artificial ant's specifically with random probability assigns each job to one processor such that each job is assigned to specific processor only. Let the artificial pheromone value be $\tau_{i,j}$ with an edge between T_i and P_j. Initially, consider that $\tau_{i,j}$ will be same for all (i, j). After each iteration, the pheromone value on the edges is updated on every trail. The behavior probability of ants in heterogeneous multiprocessor in which ants randomly chooses a node from i to j is given as [18]:

$$P\,(i,j) = \tau_{i,j} / \sum{}_{j}{}^{m}{}_{=1}{}^{*}\, \tau_{i,j} \qquad (5)$$

Also, there is a condition that more than one ant might be active at a same time. The pseudo-code of the algorithm is shown below:

Algorithm 2: ACO for scheduling problem

```
do while (solution not converged)
  for each iteration(ite)
  {
    for each ant(k)
      for each job(j)
      {
          Select the process(Pi)
          Each ant constructs solution search space
          Fitness of each ant is calculated
      }
      If schedule is feasible, compute its quality
      Select best ant on the basic of fitness
      Update the pheromone based on the quality of each
        feasible schedule.
    Evaporate the pheromone
  }
Generate the next iteration
```

The above algorithm explains the whole procedure of ant's trails using the given parameters and the scheduling criteria. These steps are more elaborated in the following discussion:

A. Solution Search Space: In order to schedule the processes, basic ACO technique is adopted. Thus the solution formed includes the formation of search space and calculation of heuristic value.

The Fig. 3 shows the outlay of search space. [12] In search space, there are m rows, n columns and n_k number processes to be scheduled, where each node represents single process. Each column consists of certain nodes which are connected to the nodes in the next column through directed edges except the n_kth node.

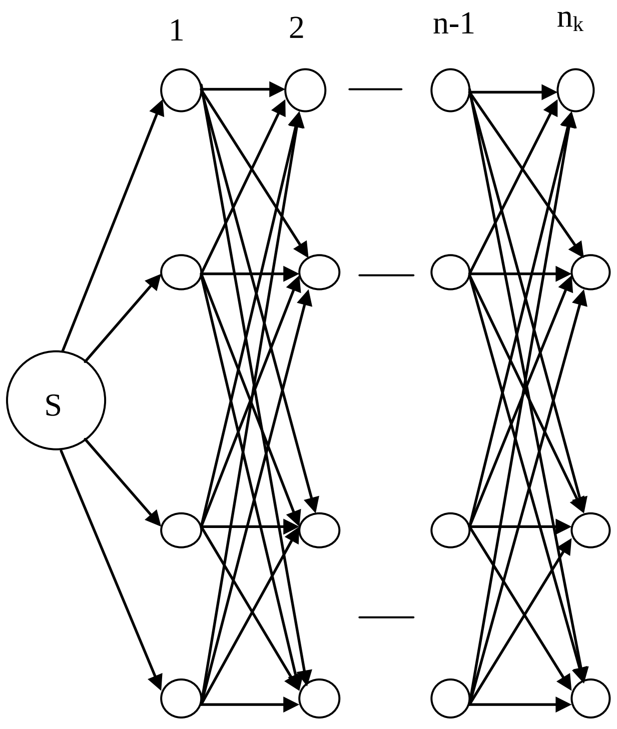

FIGURE 3. Search space

Initially ant's select the first node (process) on the basis of pheromone using roulette wheel selection method. When ants want to move to the next node, it uses the probability function, which is calculated using two parameters:

- Pheromone value on the connected edges.
- Heuristic value which is average turnaround time, average waiting time etc

B. Constraints: An ant must traverse in a sequence i.e, from left to right. A single node must be selected in each column. During the trail, ant must visit only given nodes (processes) i.e, n_k and every single node is particular in case of its label as shown in Fig. 3.

C. Heuristic: The earlier deposited pheromone trail and heuristic function influences the choice of probability. In this case for optimization based on average waiting time, heuristic function will be average waiting time of particular tour of an ant. Similarly, for optimization based on average turnaround time, heuristic function will be average turnaround time of particular tour of an ant.

D. Probability: The probability that a process will be selected in search space is calculated using two parameters i.e, heuristic value and pheromone value. In this case, probability is calculated using the following standard ACO equation:

$$P_{i,j} = (\tau_{i,j}^{\alpha} * n_{i,j}^{\beta}) / (\textstyle\sum_{k \epsilon S} \tau_{i,j}^{\alpha} * n_{i,j}^{\beta}) \quad (6)$$

Where, $\tau_{i,j}$ represents the pheromone value on edges (i , j) and $n_{i,j}$ is the heuristic value on that edges. For optimization, these values may act as scheduling criteria.

E. Roulette Wheel Selection: In this case, processes are being selected using the probability as calculated in equation (6). The chance of selection depends upon the higher probability. In roulette wheel selection method, processes are selected and assigned with area equal to their worth based on probability. This means, higher the probability of selection, larger space will be occupied.

F. Fitness Function: The fitness function calculates the fitness of the trail of each ant. The fitness of the function is based upon the optimization criteria. For example, for optimizing average waiting time, the value of average waiting time of the ants trail is used as its fitness value as shown below in equation (7):

$$\tau_{i,j} = \tau_{i,j} + fitness \quad (7)$$

$$\tau_{i,j}(t) = \tau_{i,j}(t) + q/L^{+} \quad if\ (i,j) \epsilon T \quad (8)$$

When all the ants complete a traverse, the best tour is found from the beginning of the trail (T) and quantity ($\frac{q}{L^{+}}$) where q is the constant parameter and L^{+} is the length of the best tour.

G. Pheromone: Ants navigate from food source to their colony using a chemical substance called pheromone. The two main components of ACO are pheromone updates and pheromone evaporates, as described below:

(a) Pheromone Update: The pheromone value is modified all the time by ant's navigation. After the trail is completed after a iteration, the pheromone value is updated on the path selected by the ants.[19] The equation of pheromone updation is shown below:

$$\begin{aligned} &\textit{If edge has been traversed then,} \\ &\tau_{i,j} = \tau_{i,j} + Update \end{aligned} \quad (9)$$

The quality used for pheromone updation of next iteration is given by:

$$\begin{aligned} \tau_{i,j} &= \rho * \tau_{i,j} + q(s) \quad ; \textit{if } J_i \textit{ is assigned to } P_j \textit{ in schedule S} \\ &= \rho * \tau_{i,j} \quad ; \textit{Otherwise} \end{aligned} \quad (10)$$

(b) Evaporate Pheromone: It is also known as *Pheromone decay*. After every iteration, the pheromone values on the edges gets evaporated and decayed by some set percentage. So the edge with higher pheromone concentration looses more pheromone on the edges than the edges with lower pheromone concentration. The pheromone value is evaporated by using the following equation:

$$\tau_{i,j} = \tau_{i,j} - r \quad (11)$$

Where, *r* is the decay constant which ranges from 0 to 1 exclusively. Also pheromone evaporation rule is given as:

$$\tau_{i,j} = (1\text{- } decay\ constant) * \tau_{i,j} \quad (12)$$

The parameters which are considered in this discussion are the utilization of the processor i.e, the average waiting time and average turnaround time of all the jobs and the time taken for generating the feasible schedule. For each problem instance, some trails are made to run on the processor for ACO and the average values of all the parameters are considered. These values are then compared with the scheduling algorithm and the results are tabulated. The iterations are continued till the ants come up with the definite schedule. Then the schedule is said to be converged and focalized.

VII. Results And Discussion

ACO is a metaheuristic that is probabilistic in nature, so the results thus generated will be unique if executed several times in definite number of iterations on the same problem. In this paper, results are compared and tabulated for cost function, average waiting time, average turnaround time and computational time given by ACO.

A. Shortest path problem

The ACO algorithm is implemented by completing the predefined number of iterations, whereas an ant is dropped by satisfying the predefined number of constraints before reaching its destination. ACO is used to select an optimal path along the multiple set of paths as shown in Fig (4).

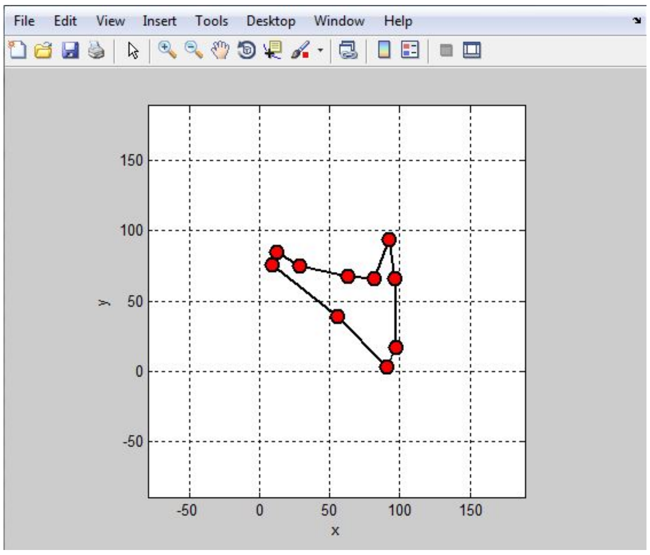

FIGURE 4. The optimal path selected by ants

Initially pheromone value is set to be 0.01. Total number of ants is set to be 20 and total number of iterations is 300. The pheromone evaporation (ρ) constant is set at 0.05.

TABLE III
DIFFERENT VALUES OF ALPHA< BETA AND RHO FOR THE OPTIMAL COST FUNCTION

S. No.	**Alpha(α)**	**Beta(β)**	**Rho(γ)**	**Cost Function**
1	0.0	0.0	0.05	319.5467
2	0.5	0.5	0.05	315.3566
3	1.0	1.0	0.05	315.3566
4	1.5	1.5	0.05	315.3566
5	1.8	1.8	0.05	315.3566
6	2.0	2.0	0.05	319.5467
7	2.5	2.5	0.05	320.2190
8	3.0	3.0	0.05	320.9161
9	3.5	3.5	0.05	319.4493
10	4.0	4.0	0.05	319.4493

Alpha, beta and rho values can range from 0.1 to 1.0. In this paper, we are trying to change the values of alpha and beta, so that we can analyze their affects on the cost function. At last, it will be clearly seen that the optimal path attained lies within the fixed range. The results thus obtained are shown in Table 3.

From Table 3, it is clear that the optimal value of cost function lies within the range of alpha, beta and rho.

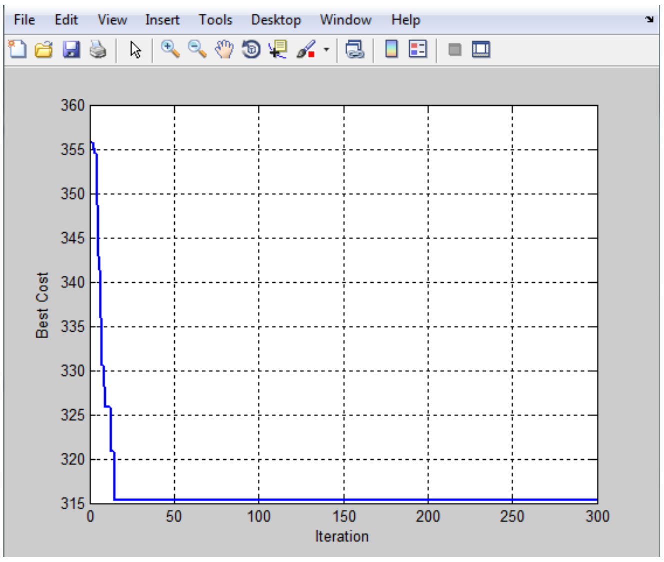

FIGURE 5. Graphical representation of cost function along the number of iteration

This calculated cost function is the fitness function which is of minimization type. The graphical representation of cost function along the given number of iterations is plotted as shown in Fig (5):

B. Average waiting time

A scheduling algorithm based on ACO is implemented and the algorithm is run for 9 problem instances with the number of processors be 9. The number of processes with their burst time is shown in Table 4. The arrival time for all the processes is same i.e, zero.

TABLE IV
BURST TIME OF ALL THE PROCESSES

Process	**Burst Time of all the processes**								
ID	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**
1	11	14	18	20	15	06	14	03	16
2	03	19	16	05	16	05	12	10	05
3	18	19	02	14	18	09	18	04	10
4	15	09	15	14	04	13	03	07	07
5	19	08	06	10	08	18	07	11	02
6	19	13	06	09	14	12	04	12	12
7	11	02	04	06	11	10	03	11	19
8	10	13	10	11	19	17	08	19	05
9	11	16	08	10	12	10	07	10	04
10	04	06	02	07	12	17	08	04	04

For scheduling algorithm, the number of ants used for ACO is 20 and the value of ρ is 0.5. Ten trails are done for each problem instance for ACO and the average values of wait time of both the algorithms i.e, FCFS and SJF are calculated and thus compared.

For each problem instance, FCFS and SJF are run with its utilization matrix (shown in Table 1) used by ACO. Thus the results are tabulated in Table 5.

TABLE V
COMPARISON OF SCHEDULING ALGORITHMS FOR AVERAGE WAITING TIME

Runs	ACO_FCFS		ACO_SJF	
	TWT	AWT	TWT	AWT
1	503	55.8889	353	39.2222
2	391	43.4444	238	26.4444
/3	498	55.3333	312	34.6667
4	411	45.6667	247	27.4444
5	405	45.0000	242	26.8889
6	436	48.4444	312	34.6667
7	247	27.4444	197	21.8889
8	448	49.7777	342	38.0000
9	400	44.4444	283	31.4444
10	240	26.6667	161	17.8889

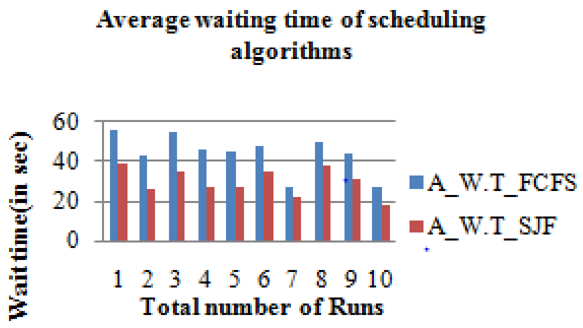

FIGURE 6. Comparison of Wait time of both the scheduling algorithms

In Table 5, it is clearly seen from the results that SJF gives the optimum results as compared to FCFS for ACO. The pictorial representation of comparison of both the algorithms for average wait time is plotted in fig (6).

Fig (6) is the clear demonstration of both the scheduling algorithm for average waiting time for given number of processes.

C. Average turnaround time

For calculating average turnaround time, consider the same file of 9 processes. There are same constant parameters i.e, alpha=1.5, Beta=1.8, rho=0.05 and ants size is 20.

TABLE VI
COMPARISON OF SCHEDULING ALGORITHMS FOR AVERAGE TURNAROUND TIME

	ACO_FCFS		ACO_SJF	
Runs	TTAT	ATAT	TAT	ATAT
1	620	68.8889	470	52.2222
2	482	53.5556	329	36.5556
3	610	67.7778	424	47.1111
4	489	55.3333	334	37.1111
5	494	54.0000	331	36.7778
6	537	59.6667	413	45.8889
7	324	36.0000	274	30.4444
8	560	62.2222	454	50.4444
9	488	54.2222	371	41.2222
10	304	33.7778	225	25.0000

If the maximum allowed iterations will be increased, average values of scheduling criteria are improved. The Table 6 shows the computed values of both the algorithms.

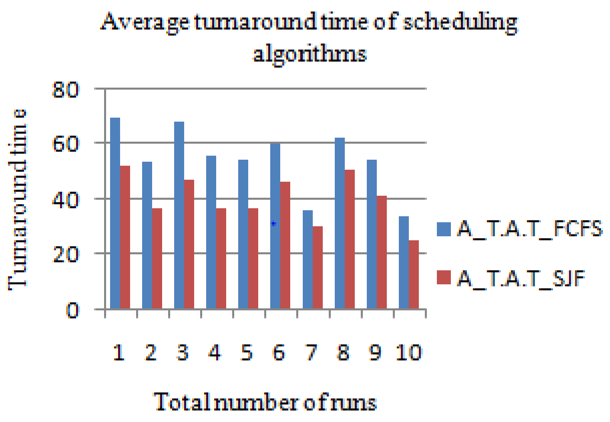

FIGURE 7. Comparison of Turnaround time of both the scheduling algorithms

In the above Table 6, total turnaround time (TTAT) is calculated for 10 trails and its corresponding average turnaround time (ATAT) for both the algorithms is also calculated. It is clearly seen from the results that are plotted in the Fig (7).

For the calculation of turnaround time, both the algorithms i.e, FCFS and SJF are calculated and the graphical representations of results are plotted as shown in the Fig above.

D. Computation time

The experimental results are shown in Table 5 and Table 6 where the comparative analysis is done between FCFS and SJF algorithm.

The comparison is done on the basis of results obtained from average waiting time and average turnaround time.

TABLE VII
COMPARISON OF SCHEDULING ALGORITHMS FOR COMPUTATIONAL TIME

Runs	ACO_FCFS (in sec)	ACO_SJF (in sec)
1	0.576943	0.626329
2	0.619968	0.591003
3	0.666770	1.396745
4	0.645646	0.821382
5	0.575617	0.786407
6	0.589067	0.647729
7	0.582400	0.756298
8	0.587648	0.678892
9	0.622608	0.727042
10	0.577861	0.883066

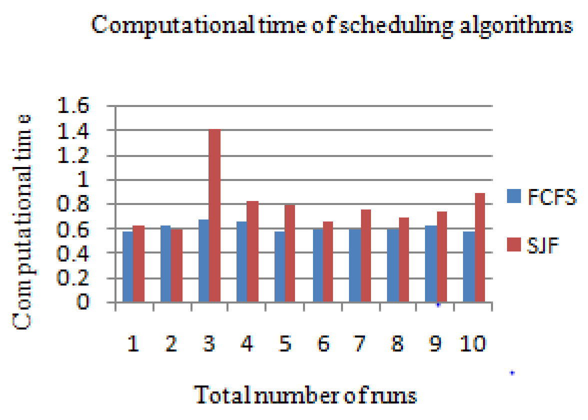

FIGURE 8. Comparison of Computational time of both the scheduling algorithms

Now the comparison is also done for the computational analysis as shown in Table 7. In the above Table 7, computational time of both the scheduling algorithm is analyzed as per run for the given number of iterations. The clear graphical representation of the above results is shown below in Fig (8).
In the above discussion, scheduling algorithms are implemented and compared using ACO for various parameters. Their comparative results are also shown using graph charts. Now various hypothesis testing is done and their corresponding results are shown in later discussion.

E. Hypothesis testing

In this paper, the primary objective of scheduling algorithms is to improve performance analysis. It depicts the usability of scheduling algorithms and compares them on the basis of different performance criteria. In order to compare the outcome of turnaround time and waiting time with different scheduling algorithms, the following hypothesis are proposed.

- Hypothesis 1:

  Ho: The turnaround time of various scheduling algorithms are significant.
  H1: The turnaround time of various scheduling algorithms are not significant.

- Hypothesis 2:

  Ho: The waiting time of various scheduling algorithms are significant.

  H1: The waiting time of various scheduling algorithms are not significant.

Case I: If *P-value <= α : Reject Ho*

Case II: If *P-value > α : Accept Ho,* there is no enough evidence to reject Ho.

E-1.1: Experimental analysis

For the testing of described hypothesis, the same scheduling algorithms are taken with sample size 10. These 10 processes are scheduled with burst time given in Table 4 and arrival time zero for every job. The turnaround time is calculated through simulator and the results are compared as shown below:

TABLE VIII
COMPARISON OF AVERAGE TURNAROUND TIME OF SCHEDULING ALGORITHMS USING ONE-SAMPLE T-TEST

	N	Mean	Std. Deviation	Std. Error Mean
A_T.A.T_FCFS	10	54.5445	11.75311	3.71666
A_T.A.T_SJF	10	40.2776	8.76434	2.77153

	Test Value = 20					
	t	df	Sig. (2-tailed)	Mean Diff.	95% Confidence Interval of the Difference	
					Lower	Upper
A_T.A.T_ FCFS	9.2 9	9	0	34.5	26.1	42.9
A_T.A.T_ SJF	7.3 1	9	0	20.2	14.0	26.5

In order to analyze the difference in performance of CPU scheduling algorithms for average turnaround time, T-test and ANOVA test are used as shown in Table 8 and Table 9.
In Table 8, one- sample T-test is applied on scheduling algorithms for analyzing the confidence level of the algorithms. On the other hand, in Table 9, significance of both the algorithms is checked by applying the ANOVA test.

TABLE IX
COMPARISON OF TURNAROUND TIME OF SCHEDULING ALGORITHM USING ANOVA TEST

A_T.A.T_ FCFS	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1017.70	1	1017.7	9.4	0.006
Within Groups	1934.54	18	107.47		
Total	2952.25	19			

In Table 9, turnaround time is significantly differ for scheduling algorithms, because the computed value (F) is highly significant at 5% level of significance and P-value is also greater the 0.05.
Hence Ho is accepted which means alternate hypothesis is rejected which proves that turnaround time of various scheduling algorithms are significant and there is no enough evidence to reject Ho.

TABLE X
COMPARISON OF AVERAGE WAITING TIME OF SCHEDULING ALGORITHMS USING ONE-SAMPLE T-TEST.

	N	Mean	Std. Deviation	Std. Error Mean
A_W.T_FCFS	10	44.2110	10.00726	3.16457
A_W.T_SJF	10	29.8555	6.95861	2.20051

	Test Value = 20					
	t	df	Sig. (2-tailed)	Mean Difference	95% Confidence Interval of the Difference	
					Lower	Upper
A_T.A.T_FCFS	7.65	9	.0	24.21	17.05	31.3
A_T.A.T_SJF	4.47	9	.002	9.855	4.877	14.8

For analyzing the difference in performance of CPU scheduling algorithms for average waiting time, T-test and ANOVA test are used as shown in Table 10 and Table 11.

TABLE XI
COMPARISON OF WAITING TIME OF SCHEDULING ALGORITHM USING ANOVA TEST

A_T.A.T_FCFS					
	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1030.39	1	1030.3	13.8	.002
Within Groups	1337.10	18	74.284		
Total	2367.50	19			

In Table 11, the computed p-value is less than 5% level of significance which means that Ho is rejected. It concludes that waiting time of various scheduling algorithms is not significant.

In the above discussion, statistical analysis and comparison of various scheduling algorithms has been analyzed. The results clearly depict the performance of existing algorithms on the basis of t-test and ANOVA test and a comparative analysis for average turnaround time and average waiting time.

CONCLUSION

In this paper, a novel n-process scheduling is done using ACO. The main approach is to find a feasible job assignment with objective to keep all the processors more or less equally loaded. On comparison of FCFS and SJF scheduling algorithms using ACO, the ACO method balances the load fairly among different process assignments. The experimental results show that the SJF_ACO gives optimal results as compared to FCFS_ACO on the basis of average waiting time, average turnaround time and computational time. The further analysis is performed for finding shortest path using ACO and a hypothesis testing is also done for scheduling algorithms to find the optimal significance of both the algorithms. The first result shows that the turnaround time of FCFS and SJF for hypothesis testing is significant and there is no enough evidence to reject Ho. The second result shows that the waiting time of FCFS and SJF for hypothesis testing is not significant which means that Ho is rejected.

In future, we will try to apply this algorithm for primitive as well as non-primitive algorithms for comparative analysis with arrival times. We will also try to apply this algorithm to find CPU throughput, utilization, response time, etc.

REFERENCES

[1]G.U.Srikanth, V.U. Maheswari, A.P. Shanthi and A.Siromoney, “A survey on real time task,” in *European J. of Scientific Research* vol. 69(1), pp. 33-41, 2012.

[2]J. Mao, “Task Scheduling of parallel programming systems using Ant Colony Optimization,” in *Proceedings of the 3rd Int. Symposium on Computer Science and Computational Technology (ISCSCT),* vol. 10,pp. 179-182, 2010.

[3]Monika and Neelam, “job scheduling using FCFS and priority queue in system,” in *int. J. of Electrical Electronics and Computer science eng.*, vol. 2, pp. 6-9, 2015.

[4]A. Abhijit, Rajguru and S.S. Apte, “A performance analysis of Task Scheduling algorithms using Qualitative parameters,” in *Int. J. of Computer app.*, vol. 74, pp. 33-38, 2018.

[5]Braun, D.T., H.J. Siegel, N. Beck, L.L. Boloni and M. Maheswaran, “A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems,” in *J. Parallel Distributed Comput*., vol. 61, pp. 810- 837, 2001.

[6]A. Radulescu and V.J.C. Gemund, “ Low-cost task scheduling for distributed-memory machines,” in *IEEE Trans. Parallel Distributed Syst.*, vol. 13, pp. 648-658, 2002.

[7]B. Ucar, C. Aykanat, K. Kaya and M. Ikinci, “Task assignment in heterogeneous computing systems,” in *J. Parallel Distributed Comput*., vol. 66, pp. 32-46, 2006.

[8]C. Blum and A. Roli, “Metaheuristics in combinatorial optimization: Overview and conceptual comparison,” in *ACM Comput. Surv.* , vol.35, pp. 268-308, 2003.

[9]C. Blum, “Ant colony optimization: Introduction and recent trends,” in *Phys. Life Rev. (IIA CSIC)*, vol. 2, pp. 353-373, 2005.

[10]Adhokshai Mishra, Ankur Verma, “Genetic Algorithm For Process Scheduling in distributed Operating System,” in *International Journal of Eng. Science and Technology*, vol. 2(9), pp. 4247-4252, 2010.

[11]Christian Blum, “Ant colony optimization: Introduction and recent trends,” *albcom, lsi, Universitat Politècnica de Catalunya, Jordi Girona1-3*, vol. 2(4), pp. 353–373, 2005.

[12]Fariha Nosheen, Sadia Bibi and Salabat Khan, “Ant Colony Optimization based Scheduling Algorithm,**”** in *Int. Conf. on Open Source System and Tech. (ICOSST), IEEE*, vol. 13, pp. 18-22, 2013.

[13]Salabat Khan, Mohsin Bilal, M. Sharif, Malik Sajid, Rauf Baig, “Solution of n-Queen Problem Using ACO,” *International Multitopic Conference, Islamabad, IEEE*, vol. 2, pp. 1-5, 2009.

[14]H. Chen and A.M.K. Cheng, “Applying ant colony optimization to the partitioned scheduling problem for heterogeneous multiprocessors,” in *ACMUSA*, vol. 2, pp. 11-14, 2005.

[15]M. Dorigo, “Optimization, Learning and Natural Algorithms,” in *Ph.D. Thesis, Politècnica di Milano*, 1992.

[16]mariusz gł ˛abowski, bartosz musznicki, przemysław nowak and piotrzwierzykowsk, “Shortest path problem solving based on ant colony Optimization metaheuristic,” in *J. of Image Processing & Communication*, vol. 17, pp. 7-18, 2013.

[17]Marco Dorigo, Vittorio Maniezzo, and Albert0 Colorni, “Ant System: Optimization by a Colony of Cooperating Agents,” *in IEEE transactions on systems, man, and cybernetics-part b cybernetics*, vol. 26, pp. 1-4, 1996.

[18]Umarani Srikanth G., V. Uma Maheswari, P. Shanthi and Arul Siromoney, "Tasks Scheduling using Ant Colony Optimization," in *J. of Computer Science*, vol. 8 (8), pp. 1314-1320, 2012.

[19]Daniel Angus, "Solving a unique Shortest Path problem using Ant Colony Optimization," in *Centre for Intelligent Systems and Complex Processes*, vol. 8, pp.1-26, 2010.

[20]Vikas Gaba, Anshu Prashar, "Comparison of processor scheduling algorithms using Genetic Approach," in *Int. J. of Advanced Research in Computer Science and Software Engineering*, vol. 2, pp. 37-45, 2012.

[21]J. Blazewicz, W. Domschkz, and E. Pesch, "The job shop scheduling problem: Conventional and new solution techniques," in *European J. of Operational Res.*, vol. 93, pp. 1-30, 1996.

[22]P. Moscato and MG. Norman, "A memetic approach for the traveling salesman problem implementation of a computational ecology for combinatorial optimization on message-passing systems," in *Int. conf. on parallel comput. And transporter application*, vol. 13, pp. 21-45, 1992.

[23]Marco Dorigo, Member *IEEE*, Vittorio Maniezzo, and Albert Colorni, "Ant System: Optimization by a Colony of Cooperating Agents," in *IEEE transactions on systems, man, and cybernetics-part b cybernetics,* vol. 26, pp. 1-4, 1996.

[24]G. D. Caro and M. Dorigo, "Extending Ant Net for best-effort quality-of-service routing," *in the Proc. of the First Int. Workshop on Ant Colony Optimization,* vol. 98, pp. 701-220, 1999.

[25]Dorigo, M. Caro, G. D., and Gambardella, L. M., "Ant Algorithms for Discrete Optimization: Artificial Life," *in the Int. J. of Advanced Manufacturing Technology*, vol. 5(2), pp. 137-172, 1999.

[26]Guoqiang Peter Zhang, "Neural Networks for Classification: A Survey," in *IEEE transactions on systems, man, and cybernetics—part c: applications and reviews*, vol. 30, pp. 127-128, 2000.

[27]K. C. Tan, T. H. Lee, D. Khoo, and E. F. Khor, "A Multi-objective Evolutionary Algorithm Toolbox for Computer-Aided Multi-objective Optimization," in *IEEE Trans. on Syst., man and cybernetics-Part B: cybernetics*, vol. 31, pp. 537-555, 2001.

[28]K. M. Passino, "Biomimicry of Bacterial Foraging for Distributed Optimization and Control," in *IEEE Control Syst. Magazine*, vol. 22(3), pp. 1-12, 2006.

[29]D. Martens, M. De Backer, R. Haesen, J. Vanthienen, M. Snoeck and B. Baesens, "Classification with Ant Colony Optimization," in *IEEE Trans. on Evol. Computation*, vol. 11, pp. 651-665, 2007.

[30]Dan Simon, Senior Member, IEEE, "Biogeography-Based Optimization," in *IEEE Trans. on Evol. Computation*, vol. 12, pp. 107-125, 2008.

[31]C. Zhang, D. Ouyang and J. Ning, "An artificial bee colony approach for clustering," in *Expert Syst. and Applications*, vol. 37 (7), pp. 4761-4767, 2010.

[32]Yang, X.S., "Firefly Algorithm, Stochastic Test Functions and Design optimization," in *Int. J. Bio-Inspired Computation*, vol. 2, pp. 78-84, 2010.

[33]Imad Zyouta, Ikhlas Abdel-Qaderb and Christina Jacob, "Embedded Feature Selection using PSO-kNN: Shape-Based Diagnosis of Micro calcification Clusters in Mammography," in *J. of Ubiquitous Syst. & Pervasive Networks*, vol. 3, pp. 7-11, 2011.

[34]Krishna H. Hingrajiya, Ravindra Kumar Gupta and Gajendra Singh Chandel, "An Ant Colony Optimization Algorithm for Solving Travelling Salesman Problem," in *Int. J. of Scientific and Res. Publications*, vol. 2, pp. 1-6, 2012.

[35]Vahid Soleimani and Farnoosh Heidari Vincheh, "Improving ant colony optimization for brain MRI image segmentation and brain tumor diagnosis, Pattern Recognition and Image Analysis," in *First Iranian conf. on pattern recognition and image analysis (PRIA), IEEE*, vol. 13, pp. 978-985, 2013.

[36]Rongali Srujana and Yalavarthi Radhika, "A Study on Recent Advances on Ant Colony Optimization Algorithm," in *Int. J. of Advanced Scientific Res. & Development (IJASRD)*, vol. 4 (02/I), pp. 89-96, 2017.

[37]Dr. Kuldeep Singh Kaswan and Amandeep, "A new technique for CPU scheduling: standard deviation based," in *Int. J. of Advanced Res. in Computer Eng. & Tech. (IJARCET)*, vol. 6, pp. 1278-1282, 2017.

[38]Jogamohan Medak and Partha Pratim Gogoi, "A comprehensive analysis of disk scheduling algorithms," in *Int. J. of Latest Trends in Eng. and Technology*, vol. 11, pp. 11-23, 2018.

[39]Sudhanshu prakash tiwari and Dr. kapil kumar Bansal, "Nature inspired algorithms on Industrial applications: A survey," in *Int. J. of Applied Eng. Res.*, vol. 13, pp. 4282-4290, 2018.

[40]Neetu Goel and Dr. R.B. Garg, "A Comparative Study of CPU Scheduling Algorithms," in *Graphics & Image Processing*, vol. 2, pp. 245-251, 2012.

Authors Profile

Ayushi received the B.E. degree in computer engineering in 2015 and the M.Tech degree in computer science in 2019 from the University of Jammu, Jammu & Kashmir. She has published two papers in international journals.

Mr. Pawanesh Abrol is professor at the Department of Computer Science & IT, and presently Head, Department of Remote Sensing And GIS, University of Jammu. He has done his Ph.D. in Computer Science from University of Jammu. Besides, he also holds the degree of MBA (HRM). Dr. Abrol has more than twenty years of teaching and research experience at post graduate level. He has received INSA visiting Fellowship to visit IIT Kanpur. He has more than forty research publications in various reputed national and international reputed international journals including SCOPUS, Thomson Reuters, (SCI & Web of Science) and conferences including IEEE. His research interests include aura based texture analysis, image analysis and authentication and eye gaze technologies. Dr. Pawanesh Abrol has been the member of various academic bodies and committees. He is also a Fellow of Institution of Engineers, Kolkata and IETE, New Delhi besides member of CSI and ISTE.