Algorithm fuzzy scheduling for realtime jobs on multiprocessor systems

Nirmala Holagundi[1], Girijamma Hollalkere Ashwathsetty[1], Mustafa Basthikodi[2]
[1]Department of CSE, RNS Institute of Technology, Bengaluru, India
[2]Department of CSE, Sahyadri College of Engineering and Management, Mangaluru, India

Article Info

Article history:

Received Aug 9, 2021
Revised Dec 28, 2021
Accepted Jan 12, 2022

Keywords:

Deadline
Fuzzification
Fuzzy logic
Inference engine
Job scheduling
Membership function

ABSTRACT

The computing in real-time is rapidly focusing much developments in technologies so that the real-time jobs are to be scheduled and executed on computing systems in particular time frame. The scheduling and load balancing techniques in distributed systems face numerous challenges because of lack of centralized strategy to dispatch the jobs in multiprocessors systems. In this work, we propose an algorithm fuzzy scheduling (AFS) for real-time jobs that includes of arrival time, deadline and computation time as the scheduling parameters of input. The approach AFS is analyzed and compared with existing fuzzy algorithm (EFA) model for evaluation of performances from the outcome of the simulation. The jobs are scheduled on multiprocessor at higher system load by making use of fuzzy mechanisms in the algorithms. The experimental results prove that the proposed AFS achieves a better performance comparatively to EFA at various system load factors with respect to mean turnaroundtime, mean response time and count of missed deadlines. This is the initial phase of the algorithm, that will be enhanced to consider a greater number of parameters to be associated with jobs for better decision making and to investigate the scope for algorithm level parallelism.

Corresponding Author:

Mustafa Basthikodi
Department of CSE, Sahyadri College of Engineering and Management
Mangaluru, India
Email: mbasthik@gmail.com

1. INTRODUCTION

With the growth of the internet and achievement of web-based apps, need for the distribution of hardware and software resources and decentralization has increased drastically, which prompted the request for latest architectures and abstractions to allow efficient exchange of resource in an administratively and geographically dispersed environment. In addition, multiprocessor platforms are extensively used, not only for servers and personal computer (PC's) but for integrated devices as well. Research on task scheduling has therefore been renewed for these multiprocessor platforms, specifically in perspective of real-time scheduling. Various multiprocessor scheduling techniques are developed and presented in the literature on the basis of single processor forecasting algorithms like gang scheduling, sharing of space and time sharing. A basic technique used in scheduling independent processes is time sharing scheduling. Space sharing is introduced in case of dependent procedures. The primary issue with this strategy is time.

In our everyday activities, we use different real-time household gadgets, but tend to know very little or nothing about it. Real-time computing and communication are needed from industrial control systems, space shuttle avionics system, mobile to missile, medical imaging systems, military systems, display systems, automated factories, traffic control systems and multiple scientific experiments. Systems scheduling has more important role in realtime systems than in non-realtime as these systems giving the correct answer very late is as terrible as not containing any answer at all. This type of system should respond to requests within a specified time that is termed deadline.

Real-time scheduling schemes includes assigning resources and central processing unit time for jobs in a way where some requirements for the performance are met. Categorically, realtime systems could be divided into two main parts that is soft and hard real-time systems. All requirements must be met in the event of hard realtime systems if not the system would be deemed to have failed, whereas certain deadlines maybe missed sometimes in soft real-time systems with only a performance loss but not a total failure. In both the systems, whenever a new task occurs, it is scheduled by the scheduler in such a way that deadline to be reached is assured. The jobs may be described as aperiodic or periodic.

In practical algorithm, the multi-processor-based scheduling contains more computational difficulty of which most researchers are unaware, this creates an opportunity for new field of research in operating systems. The two general types of multiprocessor scheduling strategies in real-time systems are: partitioning and global scheduling. Every processor under partitioning, schedule the activities independently via a local ready queue. Each task is allocated to a unique processor and is only planned and executed in that processor. In the case of global scheduling the jobs are already stored in a single queue. Whenever the scheduler is activated, the task with highest priority is selected for execution.

It’s been proved that locating a minimal schedule in the multiprocessor system for a collection of realtime jobs is NP-hard. In both cases, however, authors have done some notable contributions by these outcomes in better scheduling algorithms for multiprocessor systems. It is a difficult task to model and simulate a complicated real-world system. In order to satisfy the necessary assumptions, fuzzy control provides us with a technique for representing, manipulating and imposing heuristic knowledge for controlling a system. Fuzzy control comprises of four elements: fuzzification, inference engine, rule base, defuzzification.

The method of conversing numerical input parameters to fuzzy sets is known as fuzzification, inference engine can use them. In the real world, various hardware’s like sensors and devices, produce crisp data that is subject to a number of mistakes. Most of the quantities that we consider to be crisp, contain some ambiguity and fuzzification is generally used to tackle measurement inaccuracy. Fuzzy logic is faster; hence, it could alleviate time consumptions while making decisions [1]. The theory of fuzzy sets treats ambiguous activities mathematically and expresses degree of incomprehensibility in thinking of person by making connections to realtime number [2].

In order to achieve optimal performance, many researchers have used different fuzzy methods to schedule jobs in the recent times, however this area of multiprocessor scheduling is indeed an ongoing issue. There are several algorithms suggested in scheduling literature. With the performance and reliability trade-off of the physical subsystem, a novice periodic, fault-tolerant cyber-physical system (CPS) job model is presented [3]. Depending on the amount of consecutive previous job deadlines missed, novel scheduling technique is provided which reduces systems operations cost without giving up on stability. Scheduling of electric vehicles charging in real-world station used for charging is being formulated with sets of physical conditions to decrease the total tardiness with regard to preferred departure date [4]. To solve the problem, genetic algorithm design is used. The study done by Kumar *et al*. [5] provides a new mathematical model for allocating distributed jobs to several processors in order to obtain the best cost and system dependability. The cost of phase-wise execution, the cost of each task's residence on separate processors, the cost of inter-task communication, and the cost of each task's relocation have all been viewed as a fuzzy figure that is more realistic and accurate.

A programming tool is needed which helps to have an easy development of process of fuzzy control, containing the choice to validate the outcomes as well to tune behaviour of system [6]. In order to achieve exactness in the experimental results, fuzzy logic is used by many of the researchers, few of them to mention in literature, in data security for embedding secret messages effectively in to the medium [7] where the fuzzification is used to predict the sample space, enhancement of medical images [8] where fuzzy logic is used by stretch membership function, for the solutions to regression and classification problems [9] where author used fuzzy methods to find solutions to issues of regression dependent linguistic learning of fuzzy approaches. There are many research works make use of fuzzy logic such as enhancement of medical images [10]. Nasir *et al*. [11] proposed methodologies to find the solutions to complicated engineering challenges, that works towards obtaining and optimization of parameters of fuzzy logic control for angle tracking of hub belong to the flexibly developed manipulator systems. Generally, to change the voltage and frequency, feedback mechanisms identify processor idle time [12]. A moving mean-based scheduling of fuzzy resource is proposed for cloud environment which is virtualized, to maximize resources scheduling via virtual machines [13]. Fuzzy control is built to achieve device reachability among the specifications of the user cloud and resource availability of cloud users. Fuzzy based priority aware (FPAS) time slotted channel hopping scheduling method introduced to enable differentiation of services, to enhance life of battery and minimize latency, for determining the node's priority utilized by fuzzy logic [14].

Management of energy on side of consumers of smart grid is complicated task; it needs reliable dispatching of equipment with a reduced delay to decrease energy consumption cost and peak-to-average ratio. With heuristic optimization techniques and fuzzy logic, a hybrid optimization algorithm for schedule of appliances in home is suggested [15]. In the sensor, a probability ratio-dependent scheduling is built to smartly pick informative transmission sensors measurement along a moderate rate of transmission limit for networks that are secure [16]. For cyber-physical system (CPS) along a flexible hierarchical time-division multiple access, an arbitrated networked control system (ANCS) along with schedule actions or arbitration networks in control system suggested by experts [17].

The project work Bieza *et al.* [18] proposed single-input fuzzy logic controller to control contact pressure among pantograph catenary by implementing safe experimentation dynamics (SED) approach to tune respective variable parameters. To reduce the simple delay function and the nonlinear energy consumption function, a multiple objective-based optimization problem is developed. A scheduling-based algorithm dependent on the multiple objective-based particle swarm optimizations algorithms solves the issue. In order to obtain lower deadline missed ratios of safety based critical functionalities for automotive cyber–physical systems, an adaptive dynamic scheduling algorithm is suggested [19]. It also addresses the common challenges of criticality, parallelism, heterogeneity, safety and dynamics that are the key problems in the future generations open automotive framework. The novel intelligent algorithms i.e. An adaptive neural fuzzy inference systems (ANFIS) is introduced for neutron energy spectra unfolding to overcome the inverse challenge [20]. In order to forecast the nonlinear time series in process control with a lower count of variables, an adaptive multidimensional neuro-fuzzy inference system developed [21]. For enhancing the estimation of location in indoor environment, ANFIS is used. The residual sum of squares (RSS) values are provided as inputs to an ANFIS. Indoor and outdoor accuracy was increased by 84% and 99% for using ANFIS optimization with 0.024 m as outdoor and 0.284 m as indoor, minimum mean absolute error [22]. To identify the handcrafted set of features as embolic or non-embolic, an automated algorithm making use of ANFIS to identify cerebral embolic signal for stroke risk monitoring ANFIS is utilized. Average sensitivity of 91.5%, accuracy of 90.5% and specificity of 90.0%, is reported as important outcomes. With a two-layer decision model, a distributed optimum scheduling dependent on multi-agent system is proposed. A multistage mediation procedure on the basis of the contract net protocol employs a dynamic decision [23].

The scheduling algorithm performance under cyber physical system relies on both physical factors as well as the cyber factor. In the real scenario, both the physical factor and cyber factor in variable one is not probabilistic. For CPS scheduling for tuning the variable, soft computing is adapted [24]. Mixed criticality scheduling algorithms must be used for handling both cyber and physical parameter effectively [25]. Other criteria of the scheduling problem in CPS are adaptive energy-efficient scheduling [26]. Thus, by taking into considering all the above inputs, An ANFIS controller-based scenario analyzer as well as scheduler is recommended in this article. Hence, with the following contributions the problem is formulated.

Rajguru and Apte [27] present a fuzzy-dependent mechanism for job scheduling and balancing of loads in order to enhance distributed system's performance. In the beginning, clusters are created and node having availability of bigger buffer and higher speed of central processing unit (CPU) is selected as head of cluster. Jobs are given priorities as flexible and non-flexible by making use of strategies of job prioritization. Non-flexible jobs are made to be prioritized over the flexible jobs [28]. The parallel execution of multiple tasks in multicore environment, making use of various sets task scheduling is discussed in article [29]. The primary objective of our research is to propose better algorithm fuzzy scheduling (AFS), which minimizes the mean response time, mean turnaround time and number of deadlines missed in multiprocessors system compared to the existing fuzzy scheduling algorithms.

2. METHOD AFS MODEL

The two methods of fuzzification are: singleton fuzzifier and non-singleton fuzzifier. The fuzzifier type that is used the most is the singleton fuzzifier because it is very simple and has very little computational demands. When noise is present in data processed by the system, non-singleton fuzzifiers are used very successfully. Membership functions for the singleton and non-singleton fuzzifiers are described in (1) and (2), respectively.

$$\mu(x) = \begin{cases} 1 \text{ if } (x = x_i) \\ 0 \text{ otherwise} \end{cases} \quad (1)$$

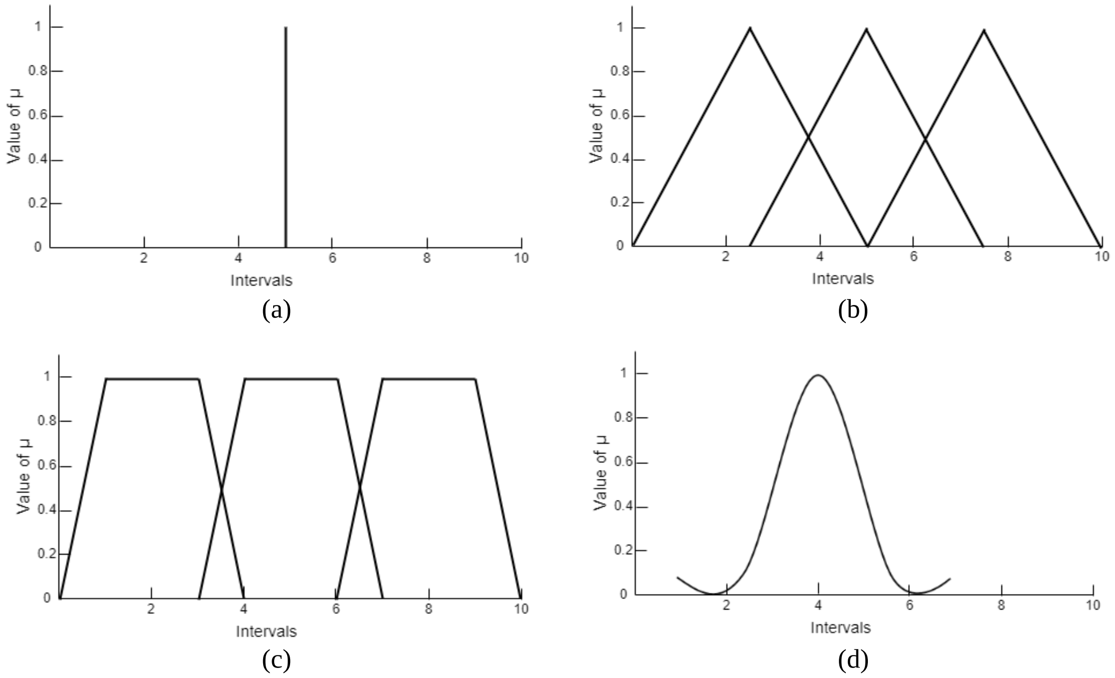
$$\mu(x) = \begin{cases} 1\ if\ x = x_{\mathrm{i}} \\ other\ otherwise \end{cases} \tag{2}$$

Theory of fuzzy sets is an expansion of crisp logic classical theory that is dependent on two truth values i.e., *True* or *False*. It is not necessary for human reasoning to use only two values. The interval between 0 *(false)* and 1 *(true)* is used by FL to provide multiple variable values. The concept of fuzzy sets is first introduced through defining a membership function and the rule, values and linguistic variables are heuristically quantified by them. In the interval [0,1], every fuzzy set member has various degrees of membership examples of a crisp set and fuzzy set is demonstrated in Figures 1(a) and (b). The Figure 1(a) give representation of set of crisp logic with two truth values. The blurred image in Figure 1(b) represents set of fuzzy logic with multiple variable values.

(a) (b)

Figure 1. Various degrees of bemberships, (a) crisp set and (b) fuzzy set; source: [30]

Input and output should be transformed to linguistic variables whose values are natural or artificial language terms, but crisp variables must be the original input and output, but intermediate procedure is a fuzzy inference process. The degree of involvement of each input value to an interval [0,1] is graphically represented in a membership function. The membership function is generally denoted as m*A* and quantifies the degree of belongingness of variable x_i to fuzzy set for each value x_i. With membership function, we illustrate how FL is utilized for measuring the significance of each linguistic definition such that the control rules defined by the application type are automated. Graphical depiction on some of the most used MF types is illustrated in Figure 2, where Figures 2(a)-(d) represents singleton MF, triangular MF, trapezoidal MF and gaussian MF respectively. Trapezoidal and triangular are most widely used because of their computational capacity and their simplicity as they are created with straight lines. Unlike other membership function, the gaussian membership function, has smooth curves but is not suitable for application that need unsymmetrical membership function.



(a) (b) (c) (d)

Figure 2. Membership functions types used, (a) singleton, (b) triangular MF, (c) trapezoidal MF, and (d) Gaussian MF

The design of fuzzy membership functions could be done with various types and sizes [31]. The process control algorithm is defined by fuzzy control, as fuzzy relationship among info on the circumstances of procedure to be controlled, *x* and *y*, performance for the process *z*. In rule base or control knowledge base, the relation between output and input is outlined in the form of guidelines. The non-fuzzy control action is generated by defuzzification operation, which best reflects MF of an inferred fuzzy controlling action. Various strategies for defuzzification methods are being proposed in literature. The four approaches that have been adapted most often among them.

Defuzzification using Tsukamoto's approach. If the monotonic MFs are utilized, then we can measure a crisp control action by:

$$Z^* = \frac{\Sigma_i^n = \omega_i x_i}{\Sigma_i^n = 1 \omega_i} \tag{3}$$

here, *n* is no. of rules along firing strength (w_i) which is higher than zero and x_i is the sum of controlling action suggested by rule *i*.

Centre of area approach. Making an assumption that controlling action having pointwise MF μ_C have been generated, the centre of area approach computes gravity centre of the distribution for the controlling action. Taking an assumption of discourse of discrete universe, we get:

$$Z^* = \frac{\Sigma_{j=1}^{q} zj\mu C(ZJ)}{\Sigma_{j=1}^{q} \mu C(ZJ)} \tag{4}$$

here *q* stands for amount of quantization levels of outcome, *j* and μ_C (z_j) describes its MF value in C, z_j is the sum of control output at quantization level.

Mean of maximum approach. The mean of maximum approach produces a crisp controlling action by taking mean of support values that exceed the maximum of their membership values. For discrete universe of discourse, its computed using (5):

$$Z^* = \sum_{j=1}^{l} \frac{zj}{l} \tag{5}$$

here *l* is no. of quantized values of *z* that meet highest memberships.

Defuzzification while rules outcome is their input function. Fuzzy control guidelines can be written as their input function. For e.g., Step-*i*: Suppose X is *Ai* and Y is *Bi* hence Z is *fi* (*X; Y*); considering a_i is firing strength of step-*i*:

$$Z^* = \frac{\Sigma_i^n = 1^{\alpha_i} f_{i(x_{i,}y_{i)}}}{\Sigma_j^n = 1 \alpha_i} \tag{6}$$

The scheduling algorithm earliest deadline first (EDF) scheduling allocated the top priority if it's having shorter deadline. The job with nearest deadline has highest priority, while the job with longest deadline is given less priority. A task due date plays a major part in first scheduling of earliest deadline and the count of jobs on the processor is set. In the parallel and distributed systems, several jobs are parallelly processed by many processors. Amount of time allocated to a processor to perform a task is called the processor's workload. Method of balancing loads between processors is regarded as load balancing.

Without the need of extended hardware, this approach achieves good efficiency. Dynamically or statically the load balancing can be achieved. On this basis, it's divided into two types, that is, dynamic and static load balancing. There are numerous problems with scheduling mechanism and load balancing in distributed since there is no centralized authorities among multiple processors for assigning the workload.

The architecture for the proposed AFS is illustrated in Figure 3, where the new fuzzy scheduler loads the set of jobs from the arrival queue into fuzzy inference engine by fuzzifying each task parameters. The fuzzy inference engine then fuzzy operator and is applied, fuzzy rules stored in the knowledge base and implication methods to generate the aggregate values which are finally defuzzified as output (runtime priority). The Scheduler follows the AFS algorithm to schedule the job and execute in the order of internal/runtime priority.

The proposed model for AFS consists of three inputs scheduling parameters arrival time, computation time and deadline. These scheduling parameters were considered because they could guarantee scheduling fairness. The outcome of system is the runtime priority which indicates the order of jobs execution in a global ready queue. Fuzzy rules join those parameters as in real worlds those are connected.

The variables as inputs are mapped into the fuzzy sets as membership functions which are demonstrated using MF diagrams in Figures 4(a)-(c) respectively. Triangular shape for the MF was used for every linguistic terms. It is very cumbersome for researcher to arrange those MFs in optimized ways. In spite of these, there exists some mechanisms for adjusting MF to be researched.

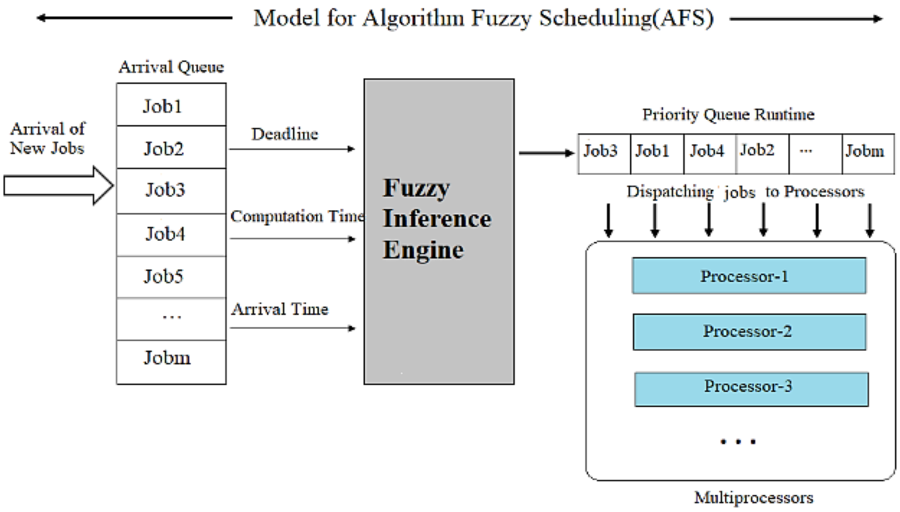
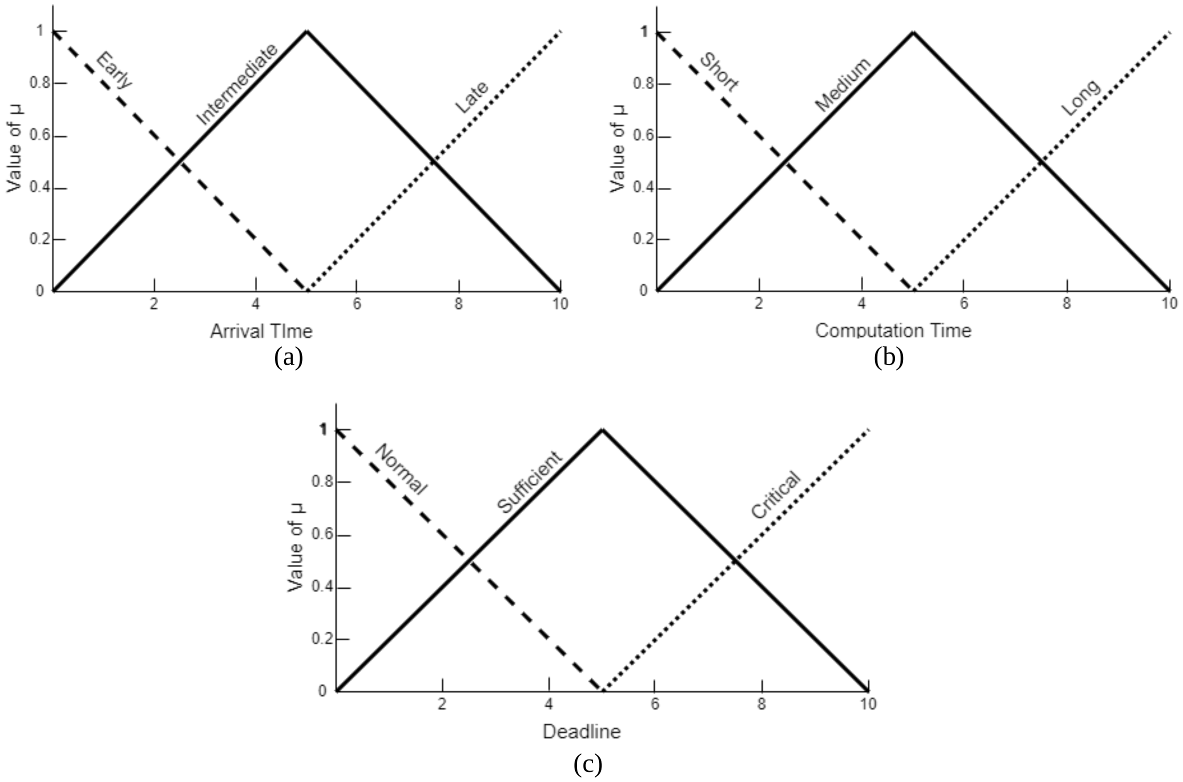


Figure 3. Model for proposed architecture of AFS



(a)

(b)

(c)

Figure 4. Membership functions for arrival time, computation time and deadline, (a) MF of arrival time for AFS, (b) MF of computation time for AFS, and (c) MF of deadline dor AFS

The twenty-seven cases of fuzzy rules used in our research work are given below, where Arrival Time (AT), Computation Time (CT) and Deadline (D), are mentioned:

Suppose (AT is early) && (CT is short) && (D is critical) then (HIGH priority);
Suppose (AT is early) && (CT is short) && (D is sufficient) then (NORMAL priority);
Suppose (AT is early) && (CT is short) && (D is normal) then (NORMAL priority);
Suppose (AT is early) && (CT is medium) && (D is critical) then (HIGH priority);
Suppose (AT is early) && (CT is medium) && (D is sufficient) then (NORMAL priority);
Suppose (AT is early) && (CT is medium) && (D is normal) then (NORMAL priority);
Suppose (AT is early) && (CT is long) && (D is critical) then (HIGH priority);
Suppose (AT is early) && (CT is long) && (D is sufficient) then (NORMAL priority);
Suppose (AT is early) && (CT is long) && (D is normal) then (LOW priority);
Suppose (AT is intermediate) && (CT is short) && (D is critical) then (HIGH priority);
Suppose (AT is intermediate) && (CT is short) && (D is sufficient) then (NORMAL priority);
Suppose (AT is intermediate) && (CT is short) && (D is normal) then (NORMAL priority);
Suppose (AT is intermediate) && (CT is medium) && (D is critical) then (HIGH priority);
Suppose (AT is intermediate) && (CT is medium) && (D is sufficient) then (NORMAL priority);
Suppose (AT is intermediate) && (CT is medium) && (D is normal) then (NORMAL priority);
Suppose (AT is intermediate) && (CT is long) && (D is critical) then (HIGH priority);
Suppose (AT is intermediate) && (CT is long) && (D is sufficient) then (NORMAL priority);
Suppose (AT is intermediate) && (CT is long) && (D is normal) then (LOW priority);
Suppose (AT is late) && (CT is short) && (D is critical) then (HIGH priority);
Suppose (AT is late) && (CT is short) && (D is sufficient) then (NORMAL priority);
Suppose (AT is late) && (CT is short) && (D is normal) then (NORMAL priority);
Suppose (AT is late) && (CT is medium) && (D is critical) then (HIGH priority);
Suppose (AT is late) && (CT is medium) && (D is sufficient) then (NORMAL priority);
Suppose (AT is late) && (CT is medium) && (D is normal) then (LOW priority);
Suppose (AT is late) && (CT is long) && (D is critical) then (HIGH priority);
Suppose (AT is late) && (CT is long) && (D is sufficient) then (LOW priority);
Suppose (AT is late) && (CT is long) && (D is normal) then (LOW priority);

In the environment of realtime processing, jobs of different characteristics are submitted to the multiprocessor by the fuzzy scheduler, this research simulate a fuzzy system comprising of real-time jobs up to 15000 jobs, which were assigned to multiprocessor based on the AFS and existing fuzzy algorithm (EFA). In order to facilitate the feasible analysis of the research, some assumptions were made. Let T_j be a periodic job, then the assumptions made are: No pre-emption of Job; All the jobs are independent; All are identical processors; Each job J_k has deadline, which is equivalent to its next period; Every Job J_k arrives to fuzzy inference engine at the same time from Arrival queue; Every job J_k becomes active in priority queue (ready queue), which arrives to the process at time zero. The fuzzy inference scheduler proposed in Figure 3 works by executing the following steps in loop as many numbers of times as required based on the arrival of jobs:
Step1: Initialization of pool of jobs M in arrival queue with job parameter $J_{k,}(AT_{k,},CT_{k,} D_k)$ for k=1,2,3…m
Step2: For every job in ready queue, input job parameter $J_{k\ ,}(AT_{k,},CT_{k,} D_k)$ into fuzzy inference engine, by considering fuzzy inference engine outcome as runtime priority for each job execution.
Step3: arrange all jobs J_k in non-increasing order as per the runtime priority into ready queue.
Step4: At the initial stage, while processors in idle state, do the following:
(a) Dispatch jobs J_k (k=1, 2,…m) with higher priority to the processors P_i (i:1, 2, ….n) and execute.
(b) Make a search through among the processors P_i (i:1, 2, …. n) which had lower computation weight.
While Comp_Weight (P_i) < Comp_Weight ($P_{(i+1)}$)
Dispatch $J_{(m+1)}$ to processor P_i to and execute
Otherwise,
Dispatch $J_{(m+1)}$ to processor $P_{\ (i+1)}$ to and execute
Step5: Make changes to state of System.

3. RESULTS AND DISCUSSION

The results of the experimentation using simulation of AFS is compared with EFA which only consists of external priority and deadline as scheduling parameters. The performance metrics used were carefully chosen in order to make reflection the real properties of a realtime systems. The performance metrics such as average response time, average turnaround time and count of missed deadlines which are most influential metrics in algorithms for scheduling of soft realtime systems.

We have considered n=10, 20, 50, and 100 as number of multiprocessors in our research during the simulation. The computation time ranges from 1-30 ms were applied across the processors and 10–15000 jobs were randomly generated with different load factors. The job parameters arrival time, computation time and deadline were generated using poisson, uniform and normal distributions respectively. Several test cases were simulated and the behaviours of both the algorithms were compared to determine the strength of the proposed algorithm. The proposed AFS outperformed the EFA [32] as demonstrated graphically in Figures 5-8 with respect to average response time.

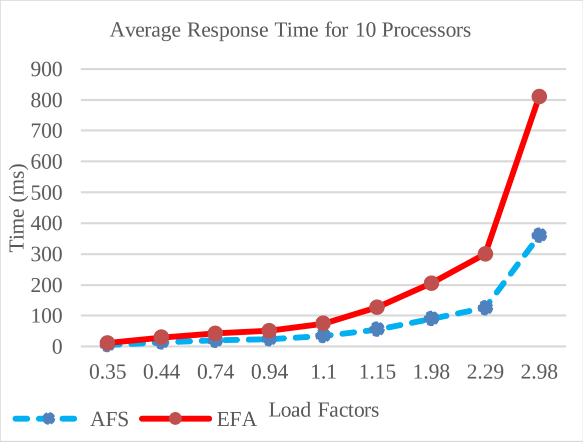


Figure 5. Analysis of average response time for n=10 processors

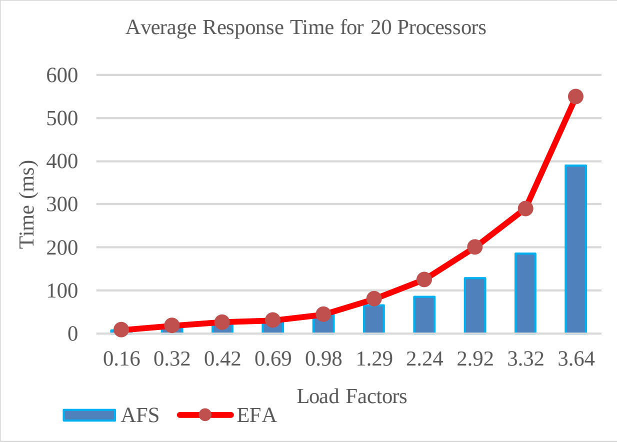


Figure 6. Analysis of average response time for n=20 processors

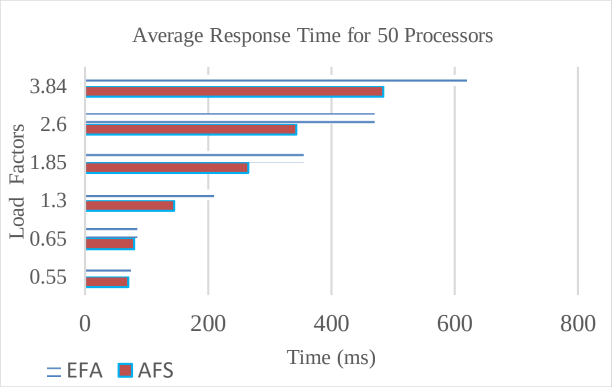


Figure 7. Analysis of average response time for n=50 processors

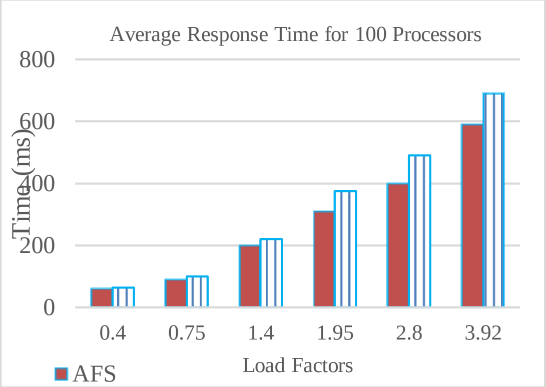


Figure 8. Analysis of average response time for n=100 processors

The graphical analysis in Figures 9-12 illustrates Average Turnaround Time of both proposed AFS and EFA, based on the different values of load factors and number of processors considered for simulation. The graphical analysis demonstrates that, when the system's load is normal, that is, when the value of load factor is below 1, both proposed AFS and EFA algorithms gives similar performance for number of processors 10 and 20. However, When the system is overloaded, that is, when the load factor become one or more, the average response time of proposed AFS starts to overtake in performance compared to EFA. The proposed AFS demonstrates much higher performance when both the load factor and number of processors increased. These results proves that the average turnaround time and average response time are minimized in our proposed AFS model.

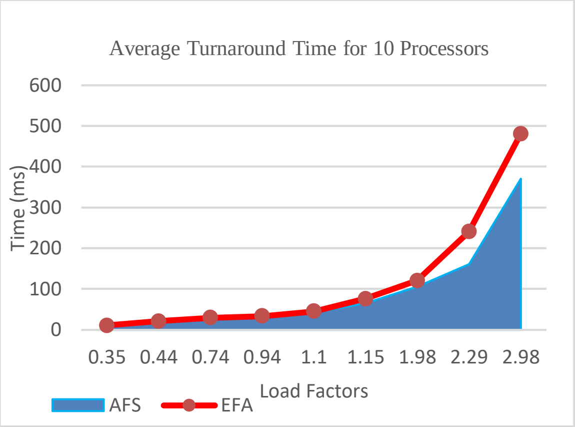


Figure 9. Analysis of average turnaroun time for n=10 processors

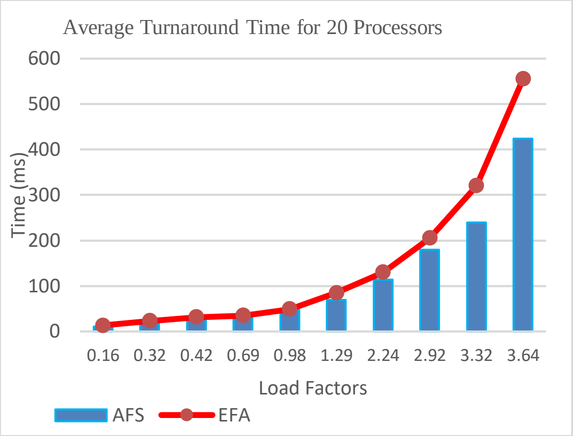


Figure 10. Analysis of average turnaround time for n=20 processors

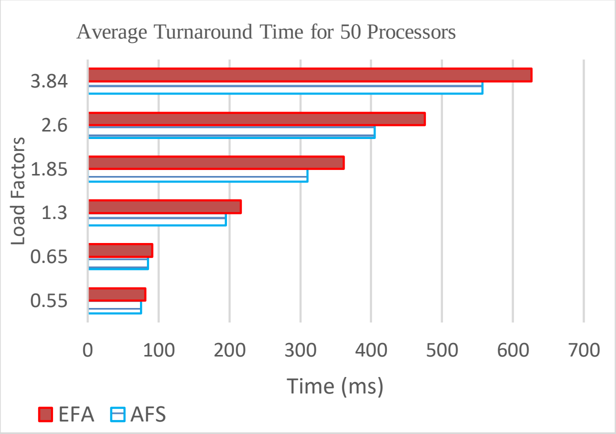


Figure 11. Analysis of average turnaround time for n=50 processors

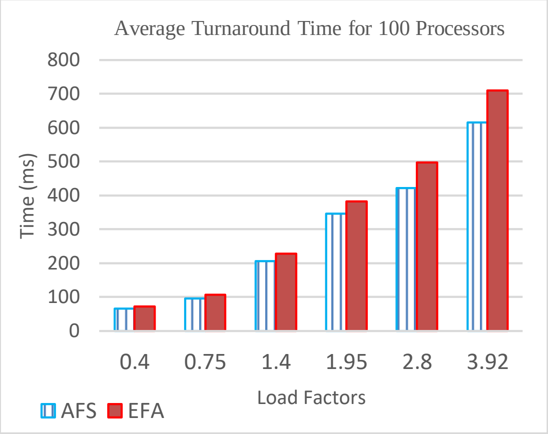


Figure 12. Analysis of average turnaround time for n=100 processors

As the system load factor becomes lower than one, the tendency for all real-time jobs is to achieve their deadline. The graphical analysis in Figures 13-16 illustrates the missed deadlines for given system load factors and number of processors considered. When the load factor is less than 1, both proposed AFS and EFA works in similar manner by achieving their deadlines. While the system load factor increases above 1, the AFS performs better than EFA, by minimizing the number of missed deadlines. T his also proves that the proposed AFS outperforms the EFA with respect to the missed deadlines in the multiprocessors system.

Discussions, the results of the proposed approach are analyzed and compared with the existing fuzzy algorithms such as evaluationary fuzzy based algorithms [32], new fuzzy scheduling algorithms [33], earliest deadline first [34] and least laxity first [35] scheduling algorithms, and proved that the proposed approach placed better compartively. The graphical analysis given above demonstrates the results gained for missed deadlines, mean response time and mean turnaround time for different load factors, by making use of different number of processors scu as 10, 20, 50, and 100, in four cases. The load facor is considered up to 3.92, when number of processors is 100, and the proposed approach considerably minimizes the response and turn aroud time. Proposed algorithm outperforms all the existing algorithms for lower and higher values of load factors in terms of mean turnaround time and mean response time for all the cases of 10, 20, 50, and 100 processors.

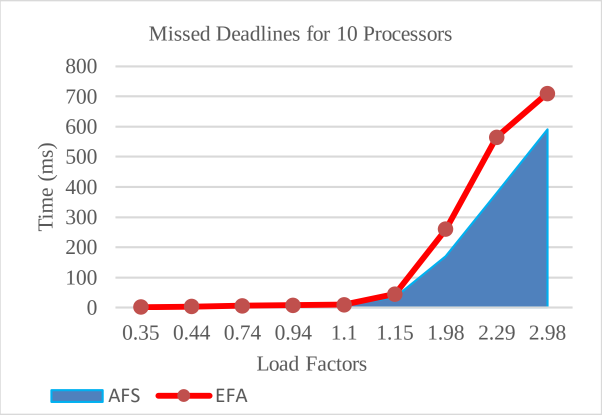


Figure 13. Analysis of missed deadlines for n=10 processors

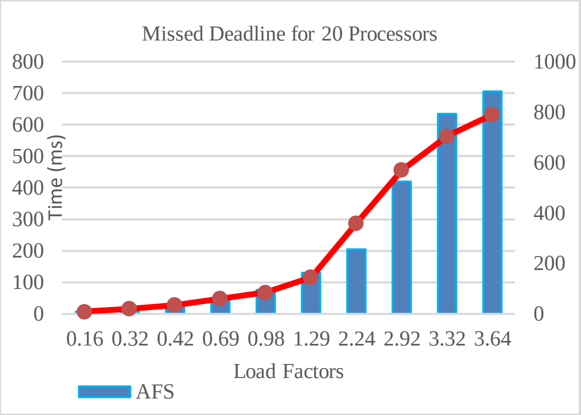


Figure 14. Analysis of missed deadlines for n=20 processors

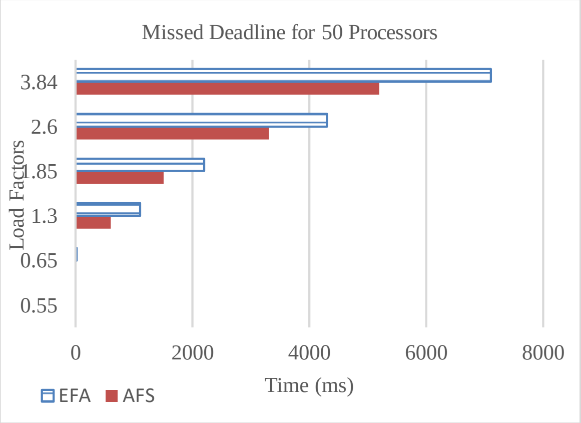


Figure 15. Analysis of missed deadlines for n=50 processors

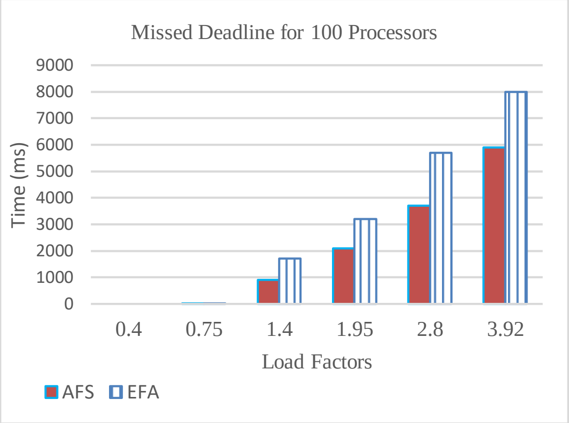


Figure 16. Analysis of missed deadlines for n=100 processors

4. CONCLUSION

In the research work carried out, the novel approach for scheduling jobs in multiprocessor systems, an AFS is proposed and experimentation is done by using simulations. We have successfully used fuzzy techniques in fuzzy inference engine to dispatch jobs to multiple processors using our approach in order to reduce average turnaround time, count of missed deadlines and average response time drastically. The experimental results achieved proved that the proposed AFS approach performs much better job scheduling in multiprocessor environment compared to existing fuzzy algorithm. As it’s initial phase of the algorithm, in the future work, the algorithm will be modified to consider more parameters in jobs such as preassigned priorities, energy consumption details, dependency between the jobs, and scopes for parallelism. The algorithm will be enhanced by introducing algorithm level parallelism by eliminating dependencies and identifying time consuming part of jobs. The advanced AFS, then will be compared and analyzed with a greater number of related works to ensure the better performance.

REFERENCES

[1] E. Danish and M. Onder, “Application of fuzzy logic for predicting of mine fire in underground coal mine,” *Safety and Health at Work*, vol. 11, no. 3, pp. 322–334, Sep. 2020, doi: 10.1016/j.shaw.2020.06.005.

[2] K. Vivek, K. V. Subbarao, W. Routray, N. R. Kamini, and K. K. Dash, “Application of fuzzy logic in sensory evaluation of food products: A comprehensive study,” *Food and Bioprocess Technology*, vol. 13, no. 1, pp. 1–29, Jan. 2020, doi: 10.1007/s11947-019-02337-4.

[3] J. Lee and K. G. Shin, “Development and use of a new task model for cyber-physical systems: A real-time scheduling perspective,” *Journal of Systems and Software*, vol. 126, pp. 45–56, Apr. 2017, doi: 10.1016/j.jss.2017.01.004.

[4] J. García-Álvarez, I. González-Rodríguez, C. R. Vela, M. A. González, and S. Afsar, “Genetic fuzzy schedules for charging electric vehicles,” *Computers and Industrial Engineering*, vol. 121, pp. 51–61, Jul. 2018, doi: 10.1016/j.cie.2018.05.019.

[5] H. Kumar, N. K. Chauhan, and P. K. Yadav, “Dynamic tasks scheduling algorithm for distributed computing systems under fuzzy environment,” *International Journal of Fuzzy System Applications*, vol. 5, no. 4, pp. 77–95, Oct. 2016, doi: 10.4018/IJFSA.2016100104.

[6] P. Sombune *et al.*, “Automated cerebral emboli detection using adaptive threshold and adaptive neuro-fuzzy inference system,” *IEEE Access*, vol. 6, pp. 55361–55371, 2018, doi: 10.1109/ACCESS.2018.2871136.

[7] M. M. Amrulloh and T. Ahmad, “Utilizing fuzzy logic in developing reversible data hiding method,” *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 5, pp. 327–336, Oct. 2020, doi: 10.22266/ijies2020.1031.30.

[8] R. A. Ali, A. M. Abbas, and H. G. Daway, “Medical images enhanced by using fuzzy logic depending on contrast stretch membership function,” *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 1, pp. 368–375, Feb. 2021, doi: 10.22266/IJIES2021.0228.34.

[9] K. Bahani, M. Moujabbir, and M. Ramdani, “Linguistic fuzzy rule learning through clustering for regression problems,” *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 3, pp. 80–89, Jun. 2020, doi: 10.22266/IJIES2020.0630.08.

[10] Y. A. Fadil, B. Al-Bander, and H. Y. Radhi, “Enhancement of medical images using fuzzy logic,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 23, no. 3, pp. 1478–1484, Sep. 2021, doi: 10.11591/ijeecs.v23.i3.pp1478-1484.

[11] A. N. K. Nasir, M. A. Ahmad, and M. O. Tokhi, “Hybrid spiral-bacterial foraging algorithm for a fuzzy control design of a flexible manipulator,” *Journal of Low Frequency Noise Vibration and Active Control*, p. 146134842110356, Aug. 2021, doi: 10.1177/14613484211035646.

[12] J. C. Charr, R. Couturier, A. Fanfakh, and A. Giersch, “Dynamic frequency scaling for energy consumption reduction in synchronous distributed applications,” in *Proceedings - 2014 IEEE International Symposium on Parallel and Distributed Processing with Applications, ISPA 2014*, Aug. 2014, pp. 225–230, doi: 10.1109/ISPA.2014.38.

[13] V. Priya and C. N. K. Babu, “Moving average fuzzy resource scheduling for virtualized cloud data services,” *Computer Standards and Interfaces*, vol. 50, pp. 251–257, Feb. 2017, doi: 10.1016/j.csi.2016.10.011.

[14] S. Kharb and A. Singhrova, “Fuzzy based priority aware scheduling technique for dense industrial IoT networks,” *Journal of Network and Computer Applications*, vol. 125, pp. 17–27, Jan. 2019, doi: 10.1016/j.jnca.2018.10.004.

[15] R. Khalid, N. Javaid, M. H. Rahim, S. Aslam, and A. Sher, “Fuzzy energy management controller and scheduler for smart homes,” *Sustainable Computing: Informatics and Systems*, vol. 21, pp. 103–118, Mar. 2019, doi: 10.1016/j.suscom.2018.11.010.

[16] J. Y. Ding, K. You, S. Song, and C. Wu, “Likelihood ratio-based scheduler for secure detection in cyber physical systems,” *IEEE Transactions on Control of Network Systems*, vol. 5, no. 3, pp. 991–1002, Sep. 2018, doi: 10.1109/TCNS.2017.2670326.

[17] H. Gong, R. Li, J. An, W. Chen, and K. Li, “Scheduling Algorithms of Flat Semi-Dormant Multicontrollers for a Cyber-Physical System,” *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 1665–1680, Aug. 2017, doi: 10.1109/TII.2017.2690939.

[18] M. F. F. Bin Kamerul Bieza, N. S. A. Shukor, M. A. Ahmad, M. H. Suid, M. R. Ghazaliv, and M. F. B. M. Jusof, “A simplify fuzzy logic controller design based safe experimentation dynamics for pantograph-cateary system,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 14, no. 2, pp. 903–911, May 2019, doi: 10.11591/ijeecs.v14.i2.pp903-911.

[19] G. Xie, G. Zeng, Z. Li, R. Li, and K. Li, “Adaptive dynamic scheduling on multifunctional mixed-criticality automotive cyber-physical systems,” *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 6676–6692, Aug. 2017, doi: 10.1109/TVT.2017.2674302.

[20] S. A. Hosseini and I. E. P. Afrakoti, “Evaluation of a new neutron energy spectrum unfolding code based on an adaptive neuro-fuzzy inference system (ANFIS),” *Journal of Radiation Research*, vol. 59, no. 4, pp. 436–441, Jul. 2018, doi: 10.1093/jrr/rrx087.

[21] J. D. Velásquez, “Adaptive multidimensional neuro-fuzzy inference system for time series prediction,” *IEEE Latin America Transactions*, vol. 13, no. 8, pp. 2694–2699, Aug. 2015, doi: 10.1109/TLA.2015.7332151.

[22] S. K. Gharghan, R. Nordin, A. M. Jawad, H. M. Jawad, and M. Ismail, “Adaptive neural fuzzy inference system for accurate localization of wireless sensor network in outdoor and indoor cycling applications,” *IEEE Access*, vol. 6, pp. 38475–38489, 2018, doi: 10.1109/ACCESS.2018.2853996.

[23] Z. Jiang, Y. Jin, M. E, and Q. Li, “Distributed dynamic scheduling for cyber-physical production systems based on a multi-agent system,” *IEEE Access*, vol. 6, pp. 1855–1869, 2018, doi: 10.1109/ACCESS.2017.2780321.

[24] N. Yi, J. Xu, L. Yan, and L. Huang, “Task optimization and scheduling of distributed cyber–physical system based on improved ant colony algorithm,” *Future Generation Computer Systems*, vol. 109, pp. 134–148, Aug. 2020, doi: 10.1016/j.future.2020.03.051.

[25] E. A. Capota, C. S. Stangaciu, M. V. Micea, and D. I. Curiac, “Towards mixed criticality task scheduling in cyber physical systems: Challenges and perspectives,” *Journal of Systems and Software*, vol. 156, pp. 204–216, Oct. 2019, doi: 10.1016/j.jss.2019.06.099.

[26] X. Dai and A. Burns, “Period adaptation of real-time control tasks with fixed-priority scheduling in cyber-physical systems,” *Journal of Systems Architecture*, vol. 103, p. 101691, Feb. 2020, doi: 10.1016/j.sysarc.2019.101691.

[27] A. A. Rajguru and S. S. Apte, “Performance improvement of distributed system through load balancing and task scheduling using fuzzy logic,” *International Journal of Engineering Systems Modelling and Simulation*, vol. 11, no. 1, pp. 35–42, Feb. 2019, doi: 10.1504/IJESMS.2019.098921.

[28] X. Xu, X. Zhang, M. Khan, W. Dou, S. Xue, and S. Yu, “A balanced virtual machine scheduling method for energy-performance trade-offs in cyber-physical cloud systems,” *Future Generation Computer Systems*, vol. 105, pp. 789–799, Apr. 2020, doi: 10.1016/j.future.2017.08.057.

[29] B. Mustafa and W. Ahmed, “Parallel algorithm performance analysis using OpenMP for multicore machines abstract : parallel,” *International Journal of Advanced Computer Technology (IJACT)*, vol. 4, no. 5, pp. 28–32, 2015.

[30] M. Cuka, D. Elmazi, K. Matsuo, M. Ikeda, M. Takizawa, and L. Barolli, “A fuzzy based simulation system for IoT node selection in an opportunistic network considering IoT node’s unique encounters as a new parameter,” in *Advances in Intelligent Systems and Computing*, vol. 1151 AISC, 2020, pp. 488–498.

[31] N. Farah *et al.*, “Fuzzy membership functions tuning for speed controller of induction motor drive: Performance improvement,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 23, no. 3, pp. 1258–1270, Sep. 2021, doi: 10.11591/ijeecs.v23.i3.pp1258-1270.

[32] H. E. Hassan, G. Nagib, and K. H. Ibrahiem, “An evolutionary fuzzy based scheduling algorithm for soft real-time multiprocessor systems,” in *Proceedings of the International Japan-Africa Conference on Electronics, Communications and Computations, JAC-ECC 2019*, Dec. 2019, pp. 174–177, doi: 10.1109/JAC-ECC48896.2019.9051189.

[33] M. T. Ogedengbe and M. A. Agana, “New Fuzzy Techniques for Real-Time Task Scheduling on Multiprocessor Systems,” *International Journal of Computer Trends and Technology*, vol. 47, no. 3, pp. 189–196, May 2017, doi: 10.14445/22312803/ijctt-v47p129.

[34] S. Zouaoui, L. Boussaid, and A. Mtibaa, “CPU scheduling algorithms: Case & comparative study,” in *2016 17th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering, STA 2016 - Proceedings*, Dec. 2017, pp. 158–164, doi: 10.1109/STA.2016.7951997.

[35] I. Khera and A. Kakkar, “Static Scheduling in Real Time for Resource Optimization,” Thesis, THAPAR University, 2012.

BIOGRAPHIES OF AUTHORS

Nirmala Holagundi obtained her Bachelor of Computer Science & Engineering from Kuvempu University Davangere, in 1998 and Master of Computer Science & Engineering from Visvesaraya Technological University Davangere, in 2005. She is currently pursuing Ph.D. and also working as Associate Professor in Department of CSE, SJBIT, Bangalore. She has published few research papers in reputed international journals and conferences including IEEE. She has 20 years of Academic experience. Her research interests include Multiprocessor scheduling Algorithms, Embedded systems, Cloud computing, and IoT. She can be contacted at email: nhnirmala13@gmail.com.

Girijamma Hollalkere Ashwathsetty obtained her Bachelor of Computer Science and Engineering from Mysore University, Master of Computer Science & Engineering from Visvesvaraya Technological University, Gulbarga and Ph. D from Kuvempu University, BIET, Davangere, India in year 2012. She is currently working as Professor in Department of Computer Science & Engineering RNSIT, Bangalore since 2008. She has published more than 30 research papers in reputed international journals and conferences including IEEE and it's also available online. Her main research work focuses on Fuzyy Automata, machine learing, Finite Automata and Formal Languages, Embedded Systems. She has 23+ years of teaching and 2 years of industry experience. She can be contacted at email: girijakasal@gmail.com.

Mustafa Basthikodi obtained his Bachelor of Engineering (B.E) in Computer Science and Engg. from Mysore University, Master of Engineering (M.E.) in Computer Science & Engineering from Bangalore university. Awarded PhD from Visvesvaraya Technological University for the research work in the area of High-Performance Computing. He is serving since last 20+ years at different capacities in academia and industries including IBM India Pvt. Ltd. At present, He is Professor in Department of Computer Science and Engg., Sahyadri College of Engineering and Management, Mangalore, India. He has published 35+ research papers in reputed International Journals and Conferences. His research interests include High Performance Computing, Multicore and Multiprocessor Technologies, Cloud Computing, Data Analytics and Machine Learning algorithms. He can be contacted at email: mbasthik@gmail.com.