IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2347-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 10, Issue 10, October 2021

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 7.569

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 10, October 2021 ||

| DOI:10.15680/IJIRSET.2021.1010106|

A Novel Adaptive Neuro Fuzzy Inference System Scheduling for Real Time Task

Mr. Sandip Ingle[1], Prof. Anil H. Rokade[2]

P.G. Student, Dept. of CSE, SYCET, Aurangabad, India[1]

HoD, Dept. of CSE, SYCET, Aurangabad, India [2]

ABSTRACT: In this propose paper a novel Adaptive Neuro Fuzzy Inference System Scheduling for to support the execution of Real Time Task based on Earliest deadline first (EDF) along with predictable applications in real time operating system. Adaptive Neuro fuzzy logic approaches are very effective for scheduling real time task. Adaptive Neuro-Fuzzy (NF) Inference System based on EDF used to execute very dynamic task. This adaptive scheduler interference considered as middle layer resources between user and RTOS. This paper shows that adaptive neuro fuzzy interference system executeEarliest deadline first tasks and optimizejob queue scheduling in Real Time operating system (RTOS). This scheduler makes decision based on 'activation function'. This proposed NF architecture, which is able to implement feed forward back propagation algorithm to job attribute present in job queueusing a set of fuzzy rules. In the literature survey discussed about various approach used for scheduling RT tasks.In the Proposed work feature of propose work and results are produce to explain performance of the proposed scheduling system.

KEYWORDS: Real Time operating system (RTOS), Adaptive Neuro-Fuzzy (NF),Earliest deadline first (EDF).

I. INTRODUCTION

Real-time operating systems (RTOSs) provide basic support for scheduling, resource management, synchronization, communication, precise timing, and I/O. RTOSs have evolved from single-use specialized systems to a wide variety of more general-purpose operating systems (such as real-time variants of Linux). We have also seen an evolution from RTOSs which are completely predictable and support safety-critical applications to those which support soft real-time applications. Such support includes the concept of quality of service (QoS) for open real-time systems, often applied to multimedia applications as well as large, complex distributed real-time systems. Researchers in real-time operating system have developed new ideas and paradigms that enhance traditional operating systems to be more efficient and predictable. Some of these ideas are now found in traditional operating systems and many other ideas are found in the wide variety of RTOS on the market today[1].A real time computer system must react to stimuli from the controlled object (or the operator) within time intervals dictated by its environment .The instant at which a result is produced is called deadline. If the result has utility even after the deadline has passed, the deadline is classified as soft, smart phones, Cameras, online transaction, data management, stock exchange etc. otherwise it is firm. If the catastrophe could result if firm deadline is missed, the deadline is hard. Commands and Control system, Command Control Systems, Airline traffic control systems, Heart Peacemaker, Airlines reservation system, Robot, Network Multimedia Systems, etc. Hard Real-Time operating system: Fig 1.1 shows real time system.

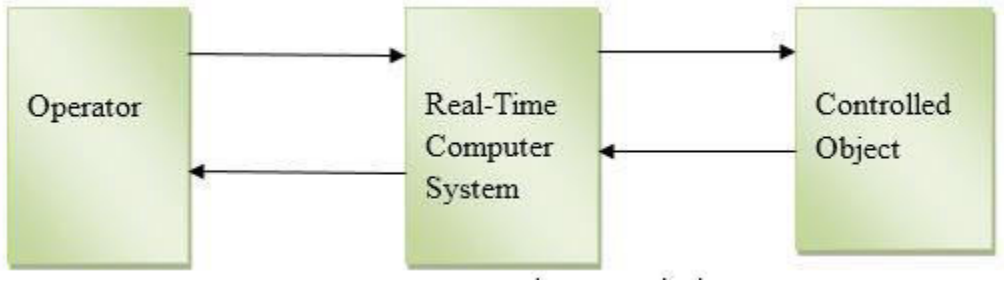


Fig. 1.1 Real-Time System

There are two main types of real-time systems: Hard Real-Time System, Soft Real-Time System.

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 10, October 2021 ||

| DOI:10.15680/IJIRSET.2021.1010106|

Hard Real-Time System requires that fixed deadlines must be met otherwise disastrous/ catastrophic situation may arise whereas in Soft Real-Time System, missing an occasional deadline is undesirable, but nevertheless tolerable. System in which performance is degraded but not destroyed by failure to meet response time constraints is called soft real time systems.

1. Types Of Operating Systems

- Real-time

A Real-time operating system is a multitasking operating system that aims at executing real-time applications. Real-time operating systems often use specialized scheduling algorithms so that they can achieve a deterministic nature of behaviour. The main objective of real-time operating systems is their quick and predictable response to events. They have an event-driven or time-sharing design and often aspects of both. An event-driven system switches between tasks based on their priorities or external events while time-sharing operating systems switch tasks based on clock interrupts.

- Neural Network

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria [3]. Neural Network is a mathematical model or computational model that is inspired by the aspect of biological neural networks. Neural network consists of an interconnected group of artificial neurons and it processes information using a connectionist approach to computation. Neural network is consider as adaptive system that will changes it structure based on the external or internal information that flow through the network during the learning phase. Neural networks are typically organized in layers. Layers are made up of a number of interconnected 'nodes' which contain an 'activation function'. as shown in th1.2. An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system

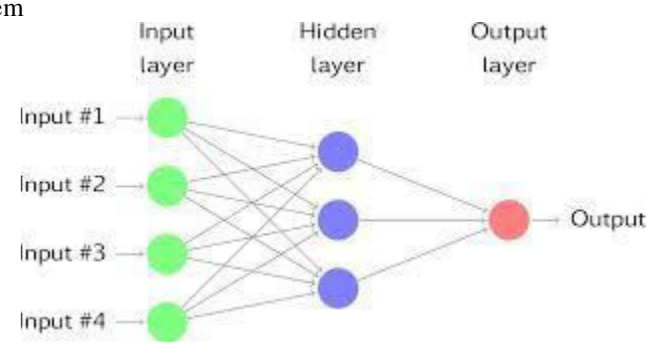


Fig.1.2: Model of Neural Network

It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons.

2. Architecture Of Neural Network

Feed-forward networks

Feed-forward ANNs permit signals to travel one way only; from input to output. There is no feedback (loops) i.e. the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straight forward networks that associate inputs with outputs. They are extensively used in pattern recognition. This type of organization is also referred to as bottom-up or top-down [14] [15].

Feedback networks

Feedback networks can have signals travelling in both directions by introducing loops in the network. Feedback networks are very powerful and can get extremely complicated. Feedback networks are dynamic; their 'state' is

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 10, October 2021 ||

| DOI:10.15680/IJIRSET.2021.1010106|

changing continuously until they reach an equilibrium point. They remain at the equilibriumpoint until the input changes and a new equilibrium needs to be found.
In Neural Network Two most common algorithm is used, Learning algorithm and back propagation algorithm. In our research we apply Back propagation algorithm. Back propagation algorithm is the most popular training algorithm. As the name implies, an error in the output node is corrected by back propagating this error by adjusting weights through the hidden layer and back to the input layer.

Real Time Task Scheduling
There are a wide variety of situations in which schedules are necessary, or at least useful. Schedules are useful for both short periods, such as a daily or weekly schedule, and for long term planning with respect to periods of several months or years. They are often made using a calendar, where the person making the schedule can note the dates and times at which various events are planned to occur. Schedules that do not set forth specific times for events to occur may instead list an expected order in which events either can or must take place.
Real time task scheduling essentially refers to determine the order in which the various tasks are to be taken up for execution by the operating system. Every operating system relies on one or more task schedulers to prepare the schedule of execution of various tasks it needs to run. Each task scheduler is characterized by the scheduling algorithm it employs. A large numbers of real time task scheduling algorithms on uniprocessors are a mature discipline now with most of the important results having been worked out in the early 1970's.[2].
Real time scheduling is divided into two types static and dynamic scheduling. In static scheduling all real time task schedule offline, using static parameter, in dynamic scheduling task scheduler calculate the feasible schedule online and allows tasks to be invoked dynamically[1]. Also it is divided into pre-emptive and non-pre-emptive scheduling. In pre-emptive scheduling each event causes interruption of running task. In non-pre-emptive scheduling a task once started is executed until completion.

Basic Terminology
Job - Unit of work scheduled and executed by system.
Task- Set of related jobs.
Periodic task- A periodic task is one that repeats after a certain fixed time interval.
Scheduler- A Module implementing scheduling algorithms.
Schedule- Assignment of all jobs to available processors, produced by scheduler.
Feasible scheduler- All task can be completed according to the set of specified constraints.
Scheduling algorithm
Scheduling means determining which tasks run when there are multiple runnable tasks. There are several computing goals that scheduling algorithms aim to fulfil. These includes-
CPU Utilization- to keep the CPU as busy as possible.
Throughput - number of processors that complete their execution per unit time.
Turnaround - total time between submission of processor and its completion.
Waiting time - amount of time a process has been waiting in the ready queue.
Response time- amount of time it takes from when a request was submitted until the first response is produced.
Fairness - Equal CPU time to each thread.
Classification of scheduling algorithm
Several schemes of classification of real time task scheduling algorithms exist.
The three main types of classification are clock driven, event driven, and hybrid.
1. Clock driven:
Table- driven
Cyclic
2. Event driven:
Simple priority-based
Rate monotonic algorithm
Earliest deadline first
Earliest deadline first (EDF): EDF algorithm optimal dynamic pre-emptive algorithm based on dynamic priorities. In this after significant event, the task with earliest deadline first is assigned highest dynamic priorities [3]. A significant event in the system can be blocking of a task, invocation of task, completion of task etc. The processor can up to 100% with EDF.
Rate monotonic algorithm (RMA): Rate monotonic algorithm is a dynamic pre-emptive algorithm based on static priorities. The rate monotonic algorithm assigns static priorities based on task periods. The task with shortest period gets the highest priority, and the task with longest period gets the lowest static priority.

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 10, October 2021 ||

| DOI:10.15680/IJIRSET.2021.1010106|

Fuzzy Logic

In the real world, information is often hazy or imprecise. When we state that it is warm today, the context is necessary to approximate the temperature. A warm day in February may be 10 degree Celsius, but a warm day in July may be 32 degrees. Human way of thinking interprets information in order to reach at conclusions. Although machines cannot yet handle inexact information in the same ways that humans do, computer programs with fuzzy logic are becoming quite useful. Fuzzy Logic is a generalization of standard logic, in which a concept can possess a degree of truth anywhere between 0.0 and 1.0. It allows intermediate values to be defined between conventional evaluations like true/false, yes/no, high/low, etc.[13].Fuzzy logic is a convenient way to map an input space to an output space. Between the input and the output we'll put a black box that does the work. Some examples where fuzzy logic is used such as automobile and other vehicle subsystems, air conditioners, cameras, rice cookers, dishwashers, elevators, washing machines etc.

Features of Fuzzy Logic

- Fuzzy logic is conceptually easy to understand.
- Fuzzy logic is flexible.
- Fuzzy logic is tolerant of imprecise data.
- Fuzzy logic can model nonlinear functions of arbitrary complexity.
- Fuzzy logic can be built on top of the experience of experts.
- Fuzzy logic can be blended with conventional control techniques.
- Fuzzy logic is based on natural language.

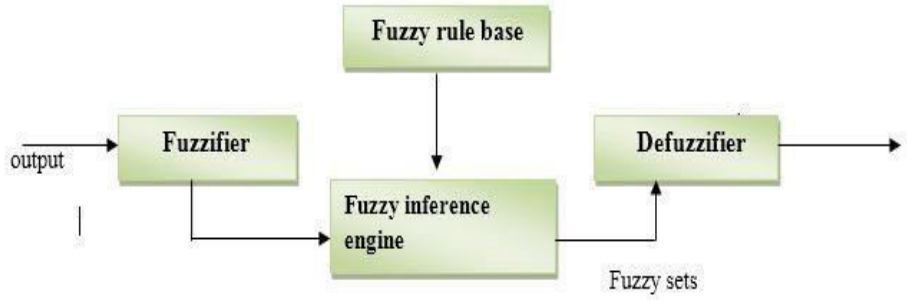


Fig.1.3: Structure of a Fuzzy System

II. LITERATURE SURVEY

In real time system various scheduling algorithms have been studied for several approaches. This chapter focuses on research of Earliest Deadline First scheduling algorithm, Neural Network and Fuzzy Logic that is most relevant to the work in this report.

In 2009, Dario Faggioli, et al [4], implemented Efficient EDF in the form of a general purpose operating system. They introduced Stander EDF scheduling policy within Linux Kernel source code, and result shown that real time task may specify a minimum inter arrival time and worst case execution time.

In 2012, Jinkyu Lee, et al. [12], proposed controlled pre-emption (CP) which controls the condition of pre-empting jobs for better EDF schedulable. The pre-emption policy determines when a higher priority job can pre-empt a currently executing lower priority job. Fully pre-emptive EDF (fp-EDF) is not always optimal. Non-pre-emptive EDF (np-EDF) is ineffective in that a higher-priority job may miss its deadline when it is blocked by a lower-priority job.

In 2013, Jag beer Singh, et al [10], have proposed Earliest Feasible deadline first (EFDF) algorithm to reduced time complexity of EDF. They analysed EFDF algorithms have least complexity. EFDF algorithm reduced the time complexity in compression of EDF algorithm on real time system scheduling for multiprocessor system.

Author Sandra G. Dykes et al. [5], proposed a system in which users define dynamic, application-specific, scheduling policies using a fuzzy-logic approach based on natural language rules. Distributed real-time applications require efficient scheduling to improve processor and network utilization to avoid missing task deadlines.

In 2012, HamidrezaRashidyKanan, et al [7], have discussed a major problem of neural network in real time applications such as long training time. To reduced neural network learning time he proposed adaptive fuzzy control system with neural network by using this system vary learning parameters and reduce training time in real time application.

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 10, October 2021 ||

| DOI:10.15680/IJIRSET.2021.1010106|

III. PROBLEM ANALYSIS

In a multiprogramming environment, the processes that are loaded into the memory compete for processor time. While a process is being executed by a processor, others wait for I/O to be performed or for some other events to take place. Scheduling, a key concept in operating system design, determines which process will progress and which will wait. Some of the objectives that scheduling function should satisfy in order to be effective include fairness, efficient use of processor time, response time, turnaround and throughput [1]. Operating systems may feature up to three distinct types of scheduling: long-term scheduling, medium-term scheduling and short-term scheduling.

IV. PROPOSED WORK DESIGN

The proposed work is based on Neural Network and Fuzzy logic. For Fuzzy logic methods we apply Mamdani-type architecture. In proposed design, first we implement feed forward back propagation algorithm to job attribute present in job queue. After that, we used methods of Fuzzy logic to job attribute which is received from NN. Mamdani-type interface expect the output membership function to the fuzzy set. After the aggregation process there is a fuzzy set for each output variable that needs defuzzification. The following figure4.1 illustrates the block diagram of proposed work.

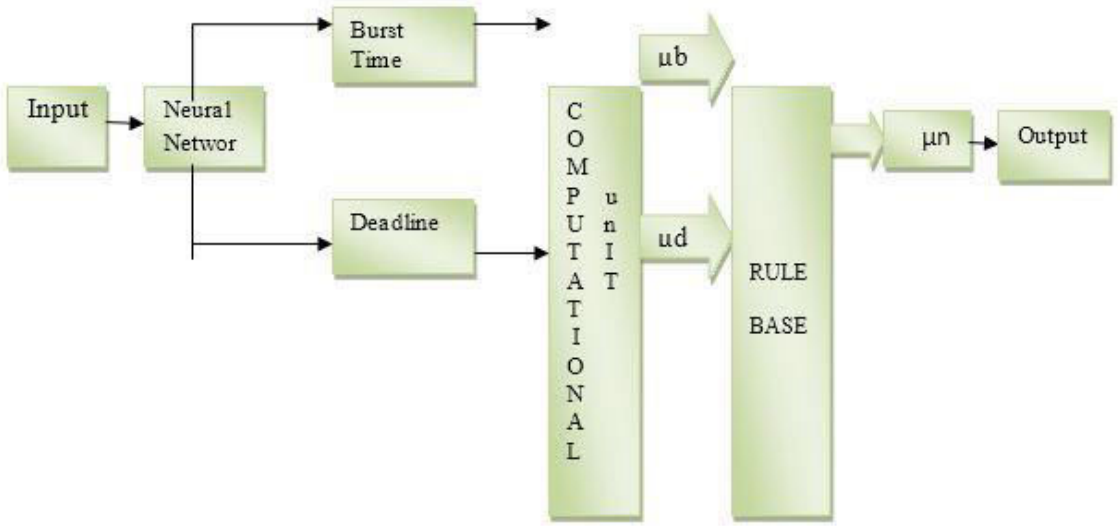


Fig. 4.1: Block Diagram of Proposed Design

In the proposed work design burst time (BT), deadline of the processes have been taken as the crisp input to the computational unit. Then membership of burst time (µb), deadline (µd) are computed. These are the input to the fuzzy rule base where new deadline of processes are evaluated individually for scheduling.

Scheduling Design using Neuro-fuzzy:

The analysed three layer feed forward back propagation NN and fuzzy logic which was proposed in our research.
The procedure used in the implementation of the scheduler is given below:

1. Began with a set of jobs (jobs are in a job queue).
2. Got basic attributes related to each job's such as burst time, arrival time and deadline.
3. Trained the scheduler with the mentioned job attributes that cover all set of possible combinations of the application environment.
4. Got the attributes of job sets from the user.
5. Trained the entered attributes with pre-calculated weights.
6. Display the job sequence based on their priorities by appending the sorted job queue and priority queue.
7. After that we apply the fuzzy logic methods in that we fuzzy the job attributes such as burst time and deadline.
8. At the last, we get optimize value of job attributes, and then schedule the task with new priority queue.

We assumed that a job is divided into different sub sets and all jobs were available in a job queue. The size of job set was equal to the size of job queue. Once the priority was assigned, a new job set was entered into the job queue. The input data sets of the neural network had 3 jobs attributes Such as burst time, arrival time and deadline.

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 10, October 2021 ||

| DOI:10.15680/IJIRSET.2021.1010106|

Job Queue

A job queue J has n possible jobs based on their arrival as shown below:

Ji = [J1, J2… Jn] Where i = 1, 2…., n

Priority Queue

The estimated job priority which was available in a priority queue P is shown below:

Priority, Pj = [P1, P2…Pn] Where j=1, 2…., n

Neuro-Fuzzy Scheduler Architecture

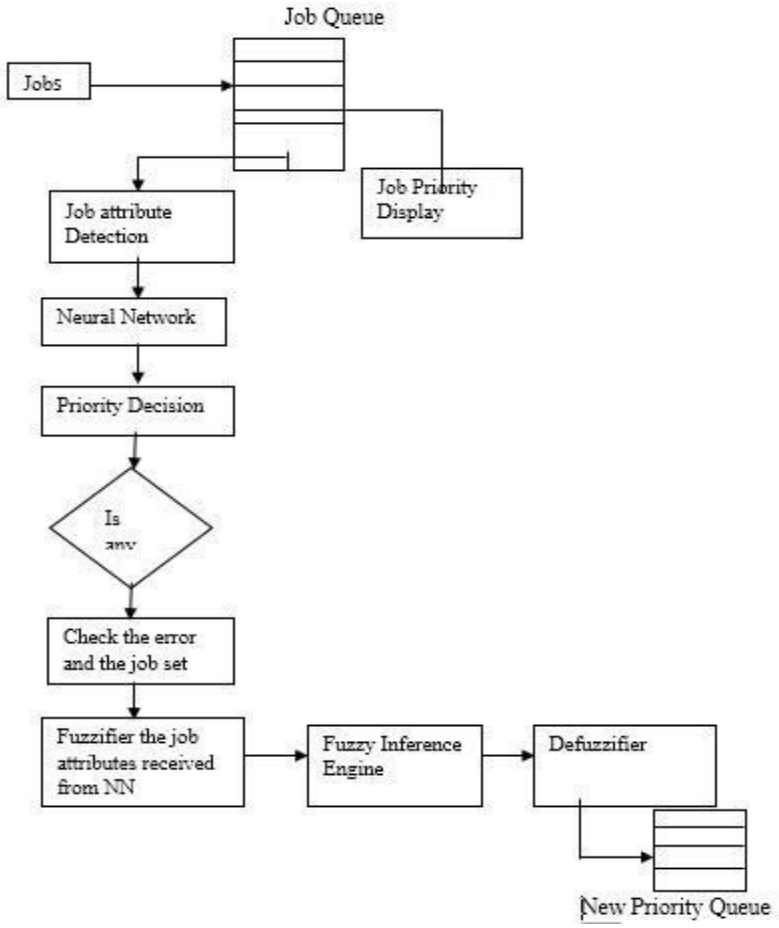


Fig 4.2 :Neuro-Fuzzy Job Scheduler Architecture

Pseudo code of the Proposed Algorithm for FL :

1. Initialize n processes with their burst time, arrival time and deadline.
2. Evaluate µd i.e. membership value of task deadline for individual processes by using the formula
µd=actual task deadline\\(maximum task deadline)+1
3 .Evaluate µb i.e. membership value of burst time for individual processes
µb=1-actual burst time\\(maximum burst time+1)
4. Find response ratios of individual processes after each iterations.
5. Evaluate µh i.e. membership value of response ratio
µh=actual response ratio/(maximum response ratio+1)
6. Evaluate µni membership value of new deadline after fuzzification for ith. Process by the formula
Pni=max{µbi, µdi, µhi}
Where 1<=i<=n
7. Apply bubble sort to get the descending order of new deadlines.

```
for(i=1;i<n;i++)
{
for(j=1;j<n-i;j++)
{
```

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 10, October 2021 ||

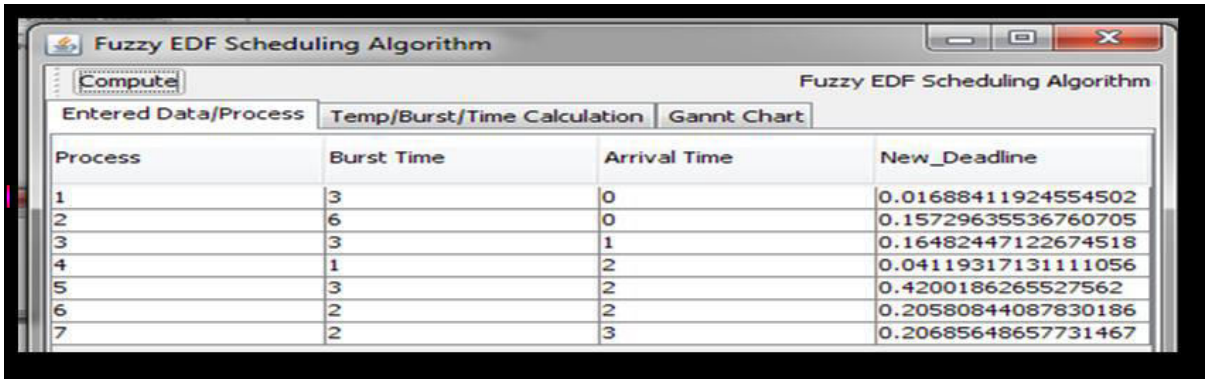
| DOI:10.15680/IJIRSET.2021.1010106|

if(Dni<Dni+1)
{
Swap the two processes
} } }
8. Execute the processes in the sorted sequence
9. Stop and Exit.

V. EXPERIMENTAL SETUP

Developed approaches are implemented using java language and eclipse. Experiments are carried out with the following hardware and software configuration.
The Platform Runtime
The primary job of the Platform runtime is to discover what plug-ins is available in the Eclipse plug-in directory. Each plug-in has an XML manifest file that lists the connections the plug-in requires. These include the extension points it provides to other plug-ins, and the extension points from other plug-ins that it requires. Because the number of plug-ins is potentially large, plug-ins are not loaded until they are actually required, to minimize start-up time and resource requirements.
The Workspace
The workspace is responsible for managing the user's resources, which are organized into one or more projects at the top level. Each project corresponds to a subdirectory of Eclipse's workspace directory. Each project can contain files and folders; normally each folder corresponds to a subdirectory of the project directory, but a folder can also be linked to a directory anywhere in the file system.
The workspace is also responsible for notifying interested tools about changes to the workspace resources. Tools have the ability to tag projects with a project nature—as a Java project, for example—and can provide code to configure the project's resources as necessary.
The Workbench
The Workbench is Eclipse's graphical user interface. In addition to displaying the familiar menus and toolbars, it is organized into perspectives containing views and editors.

Results of Fuzzy EDF scheduling algorithm for 7 processes



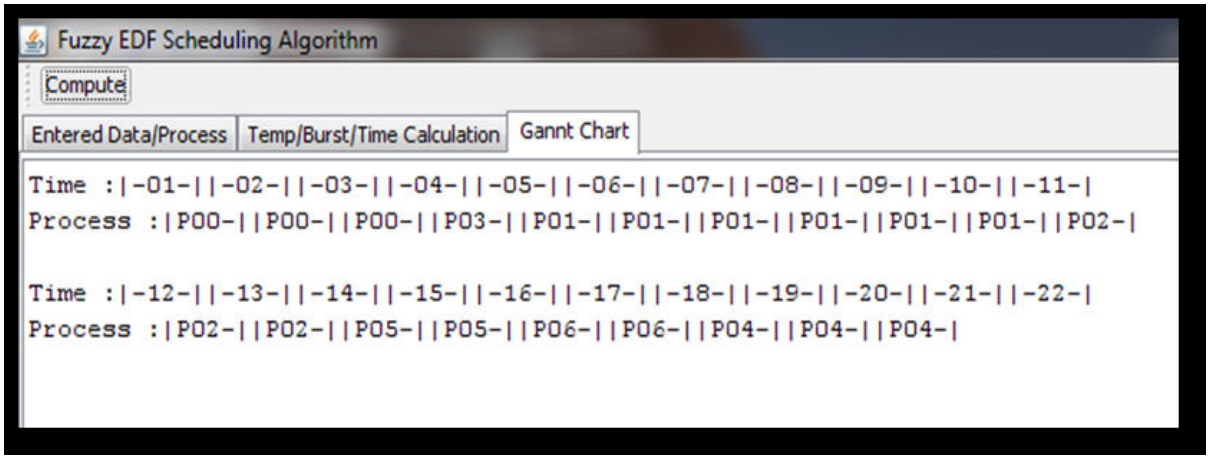
Process	Burst Time	Arrival Time	New_Deadline
1	3	0	0.01688411924554502
2	6	0	0.15729635536760705
3	3	1	0.16482447122674518
4	1	2	0.04119317131111056
5	3	2	0.4200186265527562
6	2	2	0.20580844087830186
7	2	3	0.20685648657731467

Fig.5.1 shows number of process, BT, Arrival Time of fuzzy EDF scheduling algorithm

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 10, October 2021 ||

| DOI:10.15680/IJIRSET.2021.1010106|



For EDF scheduling algorithm

Processes	5	7	9
Average AWT	4.8	8.14	16.7
Average TAT	7.6	11.0	21.5

Table 5.1 Experimental Result using EDF algorithm

For FEDF scheduling algorithm

Processes	5	7	9
Average AWT	4.0	7.47	15.6
Average TAT	6.8	10.26	20.4

Table 5.2 Experimental Result using FEDF algorithm

From above result, below shows a comparison between proposed fuzzy scheduler and EDF scheduler by graphical representation. We have done Comparisons between two important scheduler parameters average waiting time and turnaround time.

In the first bar graph, performance shows between EDF and FEDF algorithm by evaluating parameters AWT on Y-axis and no. of processes on x-axis.

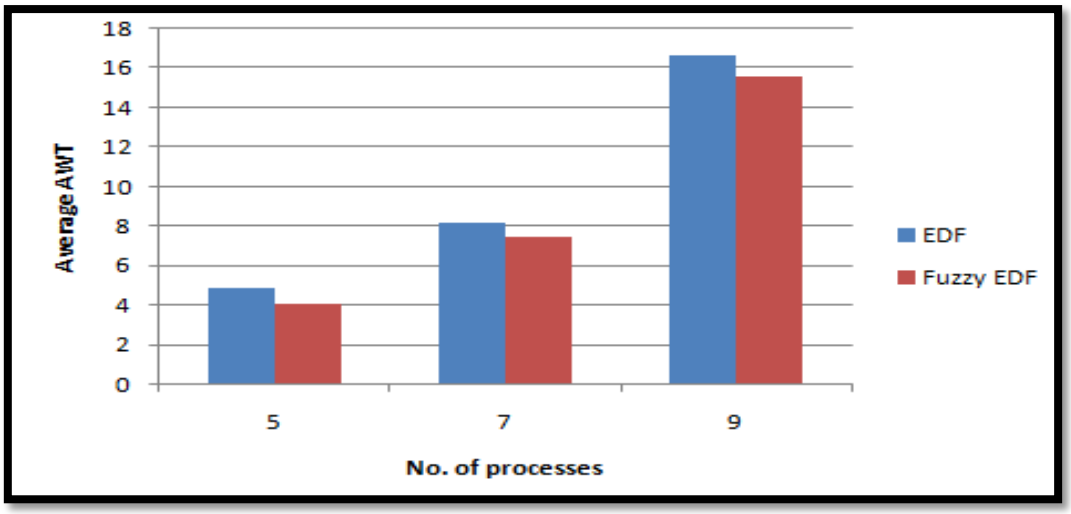


Fig.5.23 waiting time comparison between EDF and Fuzzy EDF

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 10, October 2021 ||

| DOI:10.15680/IJIRSET.2021.1010106|

VI. CONCLUSION

This paper discussed about A Novel Adaptive Neuro Fuzzy Inference System Scheduling for Real Time Task, ultimately our work motivated on neuro-fuzzy EDF scheduling algorithm.
From above experimental setup results areachieveswith following points:
1. In the proposed scheduler, introduce the neural network, fuzzy logic system with membership function and fuzzy variables.
2. From the experimental results show that our approach can reduce the average waiting time and turnaround time efficiently.
3. Produce better performance compared with the simple EDF algorithm.
Real time task scheduling is a fast getting higher application domain. It includesprogressively complex applications wherever program performancebased on time evaluation. Moreover, in these applications, even if the real time limits have to be globally respected, they are not asacute and specific as in the traditional inboard real time applications. For these explanations, it is very exciting to advance scheduler able to take into theiressential imprecision. The Fuzzy set theory is very pertinent both for capturing human constraints' formulation and imprecise task execution time evaluation. Its use to tackle real time scheduling is then very natural.

REFERENCES

[1] John A. Stankovic, et al., “Real-Time Operating Systems”, Kluwer Academic Publishers. Manufactured in The Netherlands, 2004.
[2] BinoyRavindran, et al., “Best-Effort Real-Time Scheduling Algorithms”, IEEE, Vol. 53, September 2004.
[3] Dario Faggioli, et al., “An implementation of the Eariest Deadline First algorithm in linux”, ACM, 2009.
[4] H. S. Behera, et al., “An Improved Fuzzy-Based CPU (IFCS) Algorithm for Real Time Systems”, International Journal of Soft Computing and Engineering (IJSCE) Volume-2, Issue-1, March 2012.
[5] Gil Shayer, et al., “Ranking Sensors Using an Adaptive Fuzzy Logic Algorithm”, IEEE Sensor journal, Vol. 5, No. 1, February 2005.
[6] HameshbabuNanvala, et al., “Use of Fuzzy Logic Approaches in Scheduling of FMS: A Review” international journal on computer science and engineering (IJCSE) Vol.3, No. 4, Apr 2011
[7] HamidrezaRashidyKanan, et al. ,“Reducation of Neural Network Training Time using an Adaptive Fuzzy Approach in real Time Applications”, International journal of information and electronic engineering, Vol 2, No.3 May 2012
[8] JashweeniNandanwar, et al., “An adaptive Real Time Task Scheduler”, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 6, No 1, November 2012.
[9] Jinkyu Lee, et al., “a Job or Not in EDF scheduling of uniprocessor systems”, IEEE, 2012.
[10] Kothali, Gopalkar, et al., “Neural network based priority assignment for job scheduler”, Apr 2006.
[11] Laura, et al., “Deterministic Preemptive Scheduling of Real-Time Tasks”, IEEE, 2002.
[12] Luca Abeni, et al., “On Adaptive control Techniques in Real time Resource Allocation”, Proceedings 12th Euromicro Conference on Real-Time Systems. Euromicro RTS 2000
[13] M. Kaladevi et al., “A Comparative Study of Scheduling Algorithms For Real Time Task”, International Journal of Advances in Science and Technology, Vol.1, No.4, 2010
[14] Grace. S. Wang et al., “Application Of Artificial Neural Network And Genetic Algorithm To System Identification”, Joint International Conference on Computing and Decision Making in Civil and Building Engineering, 2006.
[15] OludeleAwodele, et al., “Neural Networks and Its Application inEngineering”, Proceedings of Informing Science & IT Education Conference (InSITE) 2009

INNO SPACE
SJIF Scientific Journal Impact Factor

Impact Factor:
7.569

doi cross ref

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details