|| Volume 5 || Issue 5 || May 2020 || ISSN (Online) 2456-0774
INTERNATIONAL JOURNAL OF ADVANCE SCIENTIFIC RESEARCH AND ENGINEERING TRENDS

NOVEL APPROACH FOR REAL TIME TASK SCHEDULING USING ADAPTIVE NEURO FUZZY SCHEDULER

Pooja Madan Alone
Student, CSMSS Chh. Shahu College of Engineering Aurangabad

Abstract: - ***In a real time system contained various type of scheduling algorithms. They are used for determine which processes should be executed by the CPU when there are different processes to be executed. Adaptive Neuro fuzzy logic approaches are very effective for scheduling real time task. This paper presents scheduling algorithm of real time task using adaptive Neuro fuzzy scheduler. Then, discuss features of adaptive neuro fuzzy system. We proposed adaptive neuro fuzzy scheduler using EDF to overcome the drawbacks of other algorithms for better scheduler performance of real time task scheduling.***
Real time systems (RTS) are defence and space systems, networked multimedia systems, embedded automotive electronics, mobile computing, etc. RTS is system that must satisfy explicit (bounded) response time constraints or risk server's consequences, including failure. In other word a real time system is any information processing activity or system which has to response to externally generated input within a finite and specified period. In real time system the correctness of the system behaviour of the computations, but also on the physical instant at which these results are produced.

Keywords— *Neuro fuzzy logic, EDF, scheduler, task, process.*

I INTRODUCTION

A real time system changes its state as a function of physical time, eg. Chemical reaction continues to change its state even after its controlling computer system has stopped. Based on this real time system can be decomposed into a set of subsystem i.e the controlled object, the real time computer system and human operator. 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, otherwise it is firm. If the catastrophe could result if firm deadline is missed, the deadline is hard. Commands and Control system, Air traffic control system are examples for hard real time systems. On-line transaction systems, airline reservation systems are soft real time system. Fig 1.1 shows real time system

Man-Machine Interface Instrumentation Interface

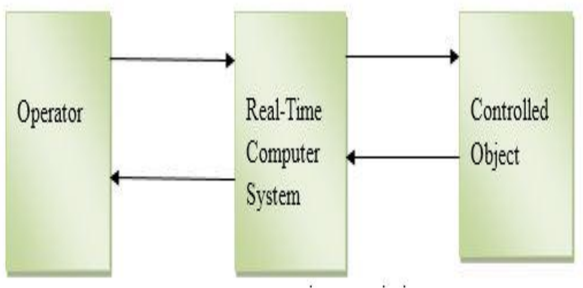


Fig. 1.1 Real-Time System

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

Operating System

An Operating system (OS) is basically an intermediary agent between the user and the computer hardware (for hardware functions such as input and output and memory allocation). It is a collection of software that manages computer hardware resources and provides common services for computer programs. The operating system is an essential component of the system software in a computer system. Operating systems can be found on almost any device that contains a computer such as cellular phones and video game consoles, super computers and web servers.

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 behavior. 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.

Multi-user

A multi-user operating system allows multiple users to access a computer system at the same time.

Single-user

operating systems have only one user but may allow multiple programs to run at the same time.

Distributed

A distributed operating system manages a group of independent computers and makes them appear to be a single computer. Distributed computations are carried out on more than one machine. When computers in a group work in cooperation, they make a distributed system.

Embedded

Embedded operating systems are designed to be used in embedded computer systems. They are designed to operate on small machines like PDAs with less autonomy.

Neural Network

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 the following figure

1.2. 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.

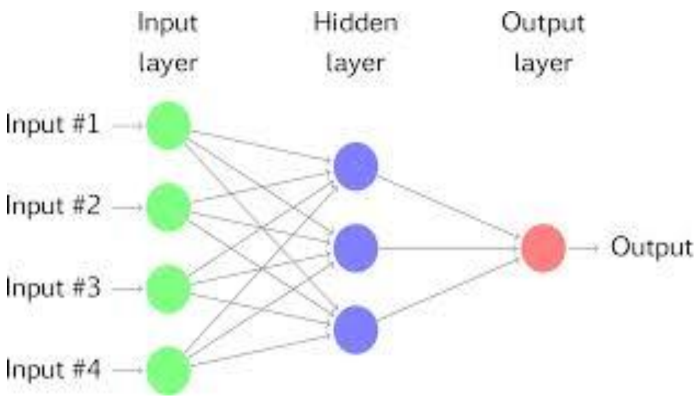


Fig.1.2: Model of Neural Network

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.

Architecture of Neural Network

Feed-forward networks

Feed-forward ANNs allow 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.

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 changing continuously until they reach an equilibrium point. They remain at the equilibrium point 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 preemptive and non-preemptive scheduling. In preemptive scheduling each event causes interruption of running task. In non-preemptive 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 fulfill. 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

3. Hybrid

Round-robin

Clock driven schedulers are simple and efficient. Therefore, these are frequently used in embedded applications. Important examples of event-driven schedulers are EDF and rate monotonic algorithm. Event driven schedulers are more sophisticated than clock-driven schedulers and usually are more proficient and flexible than clock-driven schedulers cannot. These are more flexible because they can feasibly schedule sporadic, aperiodic task and periodic task. Clock-driven schedulers can satisfactorily handle only periodic task. Out of the large number of research results that were published, the following two popular algorithms are the essence of all those results [9, 11].

Earliest deadline first (EDF): EDF algorithm optimal dynamic preemptive 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 preemptive 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.

Deadline monotonic algorithm (DM): Deadline monotonic algorithm assigns priorities to task according to their relative deadlines.

Least laxity first (LLF): LLF algorithm selects the task that has the lowest laxity among all the ready ones whenever processor become idle, and executes it to completion.

Round Robin (RR): It assigns time slices to each process in equal portions and inorder handling all processes without priority.

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.[26].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.

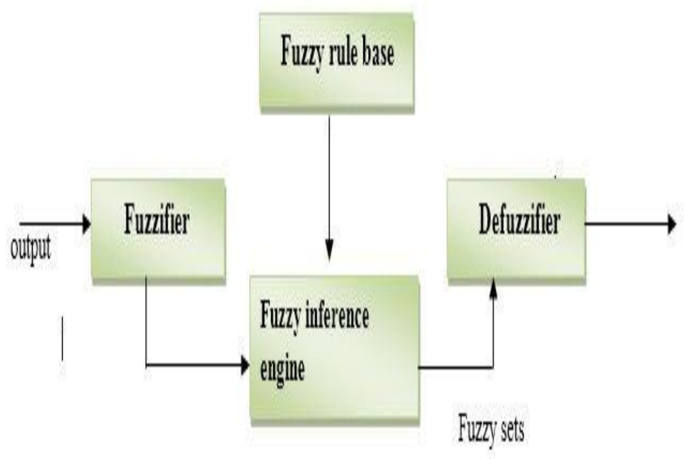


Fig.1.3: Structure of a Fuzzy System

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 2010, Xiangbin Zhu [24], proposed fuzzy control theory based on EDF algorithm. When system is overloaded misdeadlines will produce. He introduced adaptive fuzzy control scheduling algorithm.Author proposed a new EDF scheduling algorithm, which uses fuzzy control theory to reduce variance of misdeadline ratio. And show via simulation adaptive EDF algorithm has good performance when CPU utilization larger than 1.

In 2009, Dario Faggioli, Michael Trimarchi, Fabio Checconi [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, Kang G. Shin [12], proposed controlled preemption (CP) which controls the condition of preempting jobs for better EDF schedulability. The preemption policy determines when a higher priority job can preempt a currently executing lower priority job. Fully preemptive EDF (fp-EDF) is not always optimal. Non-preemptive 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, Jagbeer Singh, Bichitrananda Patra, Satyendra Singh [10], have proposed Earliest Feasible deadline first (EFDF) algorithm to reduced time complexity of EDF. They analyzed EFDF algorithms have least complexity. EFDF algorithm reduced the time complexity in compression of EDF algorithm on real time system scheduling for multiprocessor system.

In 2010, M. Kaladevi, Dr. S. Sathiyabama [16], have discussed real time scheduling algorithm and compared two important task schedulers such as EDF and Ant colony optimization (ACO) scheduler both are preemptive algorithm. Comparison based on load successive ratio and CPU utilization when load<=1, both algorithms are equally optimal. If it is compare according to execution time, EDF take less time than ACO based algorithm during under loaded condition.

In 2010, Marko Bertogna, Sanjoy Baruah [15], have proposed limited preemption real time scheduling algorithm. Because if preemption occur runtime overhead increase during execution. Preemptive EDF is an optimal scheduling algorithm for single processor systems, hence they derived limited preemption EDF scheduling algorithm to reduced runtime overhead.

In 2010, Shatha J. Kadhim et al. [21], have shown that fuzzy based scheduling algorithm useful for scheduling problems. They compared SJF, priority scheduling algorithm and fuzzy based scheduling algorithm. Next they presented via simulation comparison between parameter such as average waiting time and turnaround time in the fuzzy scheduling algorithm are better than that obtained using priority scheduling and SJF. They also proposed fuzzy decision maker which describe the relationship between measured variables and calculated output.

In 2006, Mojtaba Sabeghi et al. [17], used fuzzy logic algorithm to multiprocessor real time scheduling. Take two different fuzzy parameter, deadline and laxity. He has shown using deadline as a fuzzy parameter in multiprocessor real time scheduling is more promising than laxity.

In 2012, Sadasivam Sudha, Keppangowder Thanushkodi [22], has developed genetic algorithm based neuro fuzzy technique for process grain sized in scheduling of parallel jobs with the help of real life workload data. His result shown Genetic Based Neuro Fuzzy technique can be used as a better optimization tool for optimizing any scheduling algorithm and this optimization tool is used in agile algorithm which is used for process grain scheduling of parallel jobs.

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, Hamidreza Rashidy Kanan, Mahdi Yousefi Azar Khanian [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.

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 backpropagation 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.

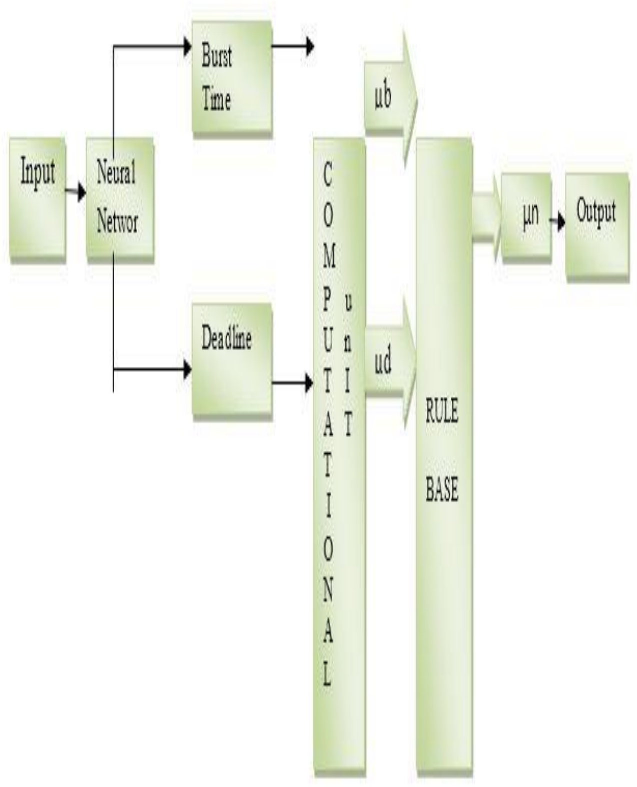


Fig. 4.1: Block Diagram of Proposed Design

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 analyzed 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.

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

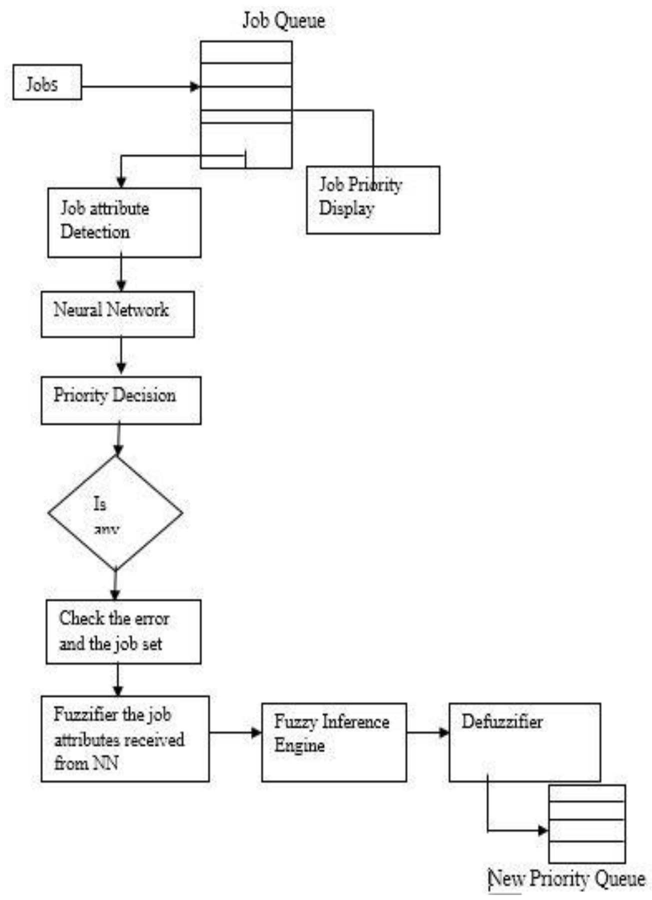


Fig 4.2 : Neuro-Fuzzy Job Scheduler Architecture

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++)
      {
             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

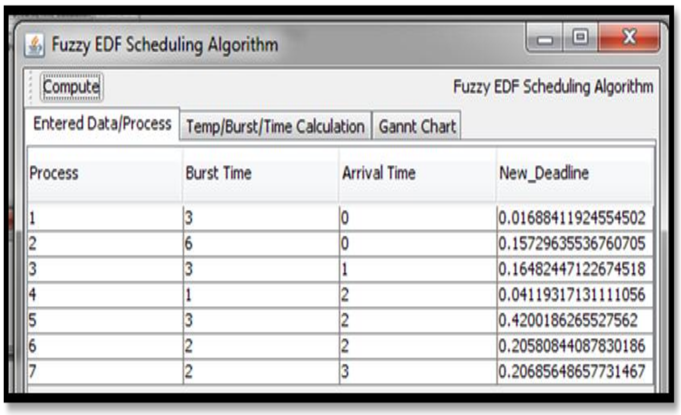
The workspace is responsible for managing the user's resources, which are organized into one or more projects at the top level. Each project corresponds to a subdirectory of Eclipse's workspace directory. Each project can contain files and folders; normally each folder corresponds to a subdirectory of 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

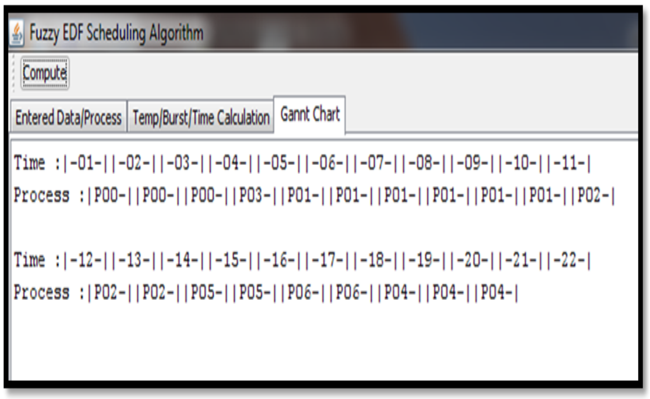
The Workbench is Eclipse's graphical user interface. In addition to displaying the familiar menus and toolbars, it is organized into perspectives containing views and editors.

Results of Fuzzy EDF scheduling algorithm for 7 processes


Fuzzy EDF Scheduling Algorithm
Compute
Fuzzy EDF Scheduling Algorithm
Entered Data/Process
Temp/Burst/Time Calculation
Gannt Chart

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



```
Time :|-01-||-02-||-03-||-04-||-05-||-06-||-07-||-08-||-09-||-10-||-11-|
Process :|P00-||P00-||P00-||P03-||P01-||P01-||P01-||P01-||P01-||P01-||P02-|

Time :|-12-||-13-||-14-||-15-||-16-||-17-||-18-||-19-||-20-||-21-||-22-|
Process :|P02-||P02-||P05-||P05-||P06-||P06-||P04-||P04-||P04-|
```

VI CONCLUSION

This report discussed about real time scheduling algorithm, basically our work focused on neuro-fuzzy EDF scheduling algorithm.

From above experimental results it concludes following points:

1. In the proposed scheduler, we applied the neural network, fuzzy logic technique with fuzzy variables and membership function.
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 expanding application domain. It involves increasingly complex applications where program execution time evaluation is very difficult. Moreover, in these applications, even if the real time constraints have to be globally respected, they are not so critical and precise as in the classical inboard real time applications. For these reasons, it is very interesting to develop scheduler able to take into account their inherent 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.

VII FUTURE SCOPE

In the future, the results of proposed fuzzy approach can be studied under different metrics with other scheduling algorithms.

Further the project can be modified by choosing more and more accurate formula for evaluating fuzzy membership value which may further reduces the waiting time and turnaround time.

REFERENCES

[1] Allalaghatta Pavan, Rakesh Jha, Lee Graba, Saul Cooper, Real-Time Adaptive Resource Management, July 2001.

[2] Binoy Ravindran, Peng Li, Fast, Best-Effort Real-Time Scheduling Algorithms, IEEE, Vol. 53, September 2004.

[3] Dario Faggioli, Michael Trimarchi, Fabio checconi, An implementation of the Eariest Deadline First algorithm in linux, ACM, 2009.

[4] H. S. Behera, Ratikanta Pattanayak, Priyabrata Mallick, 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, Ephraim Korach, and Yael Edan, Ranking Sensors Using an Adaptive Fuzzy Logic Algorithm, IEEE Sensor journal, Vol. 5, No. 1, February 2005.

[6] Hamesh babu Nanvala, 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] Hamidreza Rashidy Kanan, Mohadi fousefi , Azar khanian 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] Jashweeni Nandanwar, Urmila Shrawankar, An adaptive Real Time Task Scheduler, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 6, No 1, November 2012.

[9] Jinkyu Lee, Kang G. shin, Preempt a Job or Not in EDF scheduling of uniprocessor systems, IEEE, 2012.

[10] Kothali, Gopalkar, Neural network based priority assignment for job scheduler, Apr 2006.

[11] Laura, E. Jackson George, N. Rouskas, Deterministic Preemptive Scheduling of Real-Time Tasks, IEEE, 2002.

[12] Luca Abeni, Luigi Palopoli, On Adaptive control Techniques in Real time Resource Allocation

[13] M. Kaladevi and Dr. S. Sathiyabama, A Comparative Study of Scheduling Algorithms For Real Time Task, International Journal of Advances in Science and Technology, Vol.1, No.4, 2010

[14] Mojtaba Sabeghi, Hossein Deldari, Vahid Salmani, Malihe Bahekmat and Toktam Taghavi, A fuzzy algorithm for real time scheduling of soft periodic task on multiprocessor system, 2006.

[15] N. Audsley, A.Burns, Real Time System Scheduling, Vol.2.

[16] Rakesh jha, Bryan S. Doerr, Thomas Venturaella, Adaptive Scheduling For Real Time, Embedded information System.

[17] Runtong Zhang and Yannis A. Phillis, Admission Control and Scheduling in Simple Series Parallel Networks Using Fuzzy Logic, IEEE Transaction on Fuzzy system, Vol.9 No. 2, April 2001.

[18] Robert Fuller, neuro fuzzy methods, 2001

Society, 1988.