PERFORMANCE CONTRIBUTION BASED SCHEDULING FRAMEWORK FOR PRECEDENCE CONSTRAINT TASKS IN REAL TIME SYSTEM

RADHAKRISHNA NAIK,
SRES COE Kopargaon, Pune University,

R.R.MANTHALKAR
SGGS,COE Nanded,SRTM University Nanded,

Abstract:
It is quite possible that in a process set, process may be having earliest deadline but its performance contribution is low in process set. Scheduling such processes with highest priority does not carry any meaning. Majority of today's commercial operating system schedule task based on a single parameter however recent research on flexible scheduling showed that a single parameter is not enough to express all the application requirement .In order to provide effective support to QoS management ,an algorithm for offline scheduling of communicating tasks with precedence constraints suggested on uniprocessor Since in real time system processes are communicating to each other, if any message is failed then performance of executing successor process is reduced up to some extent .Contribution of each process in process network can be evaluated offline using multistate system (MSS) analysis. This paper suggests the policy to convert task precedence and communication constraints into pseudo deadlines of task. This scheduling policy is compared with latest deadline first (LDF).Major advantage of our policy is to support cyclic process network. Where as LDF supports acyclic process network.

***Keywords**: PCF: Performance contribution factor, MSS: Multi state system, DAG: Directed acyclic graph, FDL: feature descriptive language.*

I. INTRODUCTION

In a typical real time system, tasks interact directly or indirectly with each other. Tasks may interact in order to synchronize their execution by a message transmission or they may share resources other than processor. These resources may be exclusive or shared form. This creates precedence relationship among tasks. Precedence relationship is known before execution i.e. they are static and can be represented by a static precedence graph. If a process is not ready, but its output is necessary for the execution of next process then next process has to wait for the execution irrespective of its priority. However it is not possible to keep the processor idle for that time. Therefore it is essential to consider process dependency while considering scheduling.

In spite of the increased system complexity real time application are mainly configured acting on the task priorities, which usually express the importance of task [1].There are other system constraints like message communication .Which need to map into a set of priority levels. Another problem with priorities are those activities which often consist of several tasks, which may play different roles in different scenarios , making the priority assignment even more difficult [2].In today's scenario scheduler should be reflective i.e. it should reflect the application characteristics into a set of parameter which can be used by appropriate scheduling algorithm to optimize system performance. For example typical parameters that may be useful for effective task management include deadlines, priority constraints, importance, QoS values (contribution factor) and so on.

It is possible that, different type of possibilities and runtime changes can be included in system analysis.

Modern large scale real time systems are distinguished by their structural complexity. Many of them can perform their task at several different levels. In such cases, the system failure can lead to decreased ability to perform the given task, but not complete failure. In addition each system element can also perform its task with some different levels. The physical characteristics of the performance depend on the physical nature of system outcome. Therefore it is important to measure performance rates of system components i.e. process by their contribution into the entire MSS output performance.

Consider greenhouse monitoring and control system in which one module is intended for sensing various temperature, humidity, light etc. Another module is for controlling module .Control module intern actuates various devices and display module displays the temperature, humidity, light etc. as shown in fig. 1.

This set of four modules can be translated into a task composed of four tasks with precedence and communication constraints. Note we have assumed that each task executes sequential code and communication between two tasks . We mean that communication that occurs when one completes execution and sends its results to other before the later begins execution.
This algorithm consists of two parts:
1. Classification of tasks based on their PCF and relative deadline in various classes.
2. Allocation of tasks to processor.

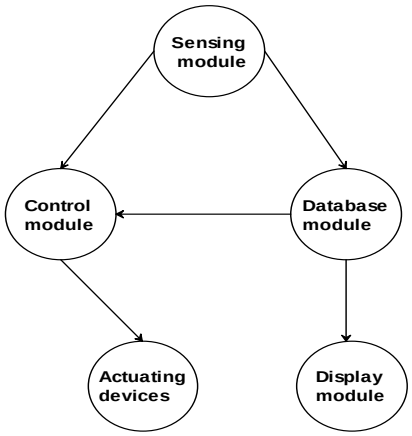


Figure: 1: Performance contribution of each process in given network

In this paper, we deal with analysis of communication network and try to find out performance contribution of each process with respect to each other. After identifying process contribution in terms of message communication, they are classified by considering both process contribution and deadlines. Pre runtime allocation of communicating periodic tasks to processor is designed. By "allocation" we mean assignment with subsequent scheduling considered. The objective of our algorithm is to decide that whether or not it is possible to schedule tasks under the given assignments such that all of their deadlines and precedence constraints can be met. This is in contrast to conventional methods which deal with either assignment or scheduling of tasks , alone but not both.
Following are features of our algorithm:
- Tasks communicate with each other during the course of their execution to accommodate a common system goal.
- The tasks to be allocated are invoked periodically at fixed time intervals during the mission lifetime.
- Consequences of failure link on other process are calculated.
- Accordingly process rank is decided.

II. RELATED WORK

A. Use of modeling in predictability analysis:

Modeling plays a central role in system engineering .Modeling methodologies should be closely related to implementation methodologies for building correct real time system .These methodologies should support end to end constraints at step in the design process [2].

Modeling system in the large is an important research topic in both academia and industry [2]. A key issue in a modeling methodology is the issue of adequate operators to compose heterogeneous schedulers (e.g. synchronous, asynchronous, event triggered or time triggered). For this reason, some researchers propose model based theories for computing scheduling policies [3]. Another challenge consists in adequately relating the functional and non functional requirements of the application software with the underlying execution platform .These are two current approaches to this problem.
1. One relies on architecture description languages that provide means to relate software and hardware components .e.g. meta-H [4].
2. The other is based on the formal verification of automata based models automatically generated from software and appropriately annotated with timing constraints [5, 6]. When a process fails it means it may

contribute none or up to some accomplishment level. Inter process message communication affects task schedulability and thus, has to taken into account.

B. Issues involved in precedence constraint task scheduling

Since the problem of assigning tasks subject to precedence constraints is generally NP hard [7, 8, 9, 10].Hence in practice it is not possible to determine optimal schedules efficiency some form of approximation using heuristics needs to be developed for this problem .For example CP/MISF (Critical Path / Most Immediate Successors First), enumeration tree of task is generated and searched using a heuristic algorithm [11]. Chu and Leung [12] presented an optimal solution to the task assignment problem in the minimizing average task response time subject to certain timing constraints. Shen and Tsai[13],Ma *et al*. [14] and Sinclair[15] derived optimal task assignment to minimize the sum of task execution and communication costs with the branch and bound[16].Since embedded system's complexity is growing day by day software and hardware requires an automated resource allocation automatically .For example, task allocation algorithm have been suggested for process control [17, 18] turbo engine control [19].The complexity of the allocation problem usually calls for the use of heuristic solutions.

Latest Deadline First Algorithm (LDF) suggested by L.Lawler[20] is as below-

1. Among tasks without successors select the task with the latest deadline
2. Remove this task from the precedence graph and put it into a stack
3. Repeat until all tasks are in the stack
4. The stack represents the order in which tasks should be scheduled
5. LDF is optimal.

Although LDF is optimal algorithm however it support only acyclic graph it does not support cyclic graph for analysis.

III. MAJOR ISSUES INVOLVED IN SCHEDULING

Since the first results published in 1973 by Liu and Layland [1] on the Rate Monotonic (RM) and Earliest Deadline First (EDF) algorithms, a lot of progress has been made in the schedulability analysis of periodic task sets. Unfortunately all analysis is done based on schedulability, jitter, number of preemptions, runtime overhead, and robustness during overloads, the transient overload etc. However today hardware technology is improved. Hardware resources are cheaper and speed of hardware has increased drastically. Also complexity of real time system is increased today.

System effectiveness can be defined more precisely as the expected value of its worth as identified for a given benefit in given time. No matter how quick a process has a deadline, its overall impact in process structure in terms of its performance contribution in process network need to be evaluated so that one can decide its priority for scheduling.

- It is also important to consider dependability of tasks. In a complex real time system, very few tasks are independent of each other. A scheduling algorithm which will work for dependable periodic tasks is needed to be implemented.
- While evaluating performability of system, one major issue is whether to analyze system after deployment, i.e. black box based, or analyzes system before deployment. As far as real time systems are concerned, deploying system on site and studying its behavior is practically not feasible.
- When a process fails means it may contribute none or up to some accomplishment level. Since tasks are getting executed on behalf of each process. It means that when a process fails means none of tasks executed or some tasks are executed. Is it possible that to estimate this performance contribution while designing a scheduler?

IV. PROPOSED FRAMEWORK

A. Overview of the algorithm

Step I

Various modules and processes in each module are identified. Signals between processes are also identified. Resources required for the system are identified.

Step II

System's behavior is presented using FDL in EVENT STUDIO.

Step III

Process interaction collaboration diagram is generated. It represents process communication diagram.

Step IV
Simplified process interaction diagram is evaluated from process interaction collaboration diagram.
Step V
Precedence of processes is identified and conditional probability is calculated. If a process is dependent on two or more processes, then Bay's theorem for conditional probability is used to find out dependability of each process with others.
Step VI
Accomplishment level for each process is assumed, based on criteria that how many incoming links are associated with each process.
Step VII
Performance contribution factor for each process is calculated.
Step VIII
Processes are classified on PCF and relative deadline to four classes, class-I, class-II, class-III and class-IV. While scheduling processes, highest priority is given to class with Highest CF and quickest deadline.
Block diagram of proposed algorithm is explained in Fig. 2.

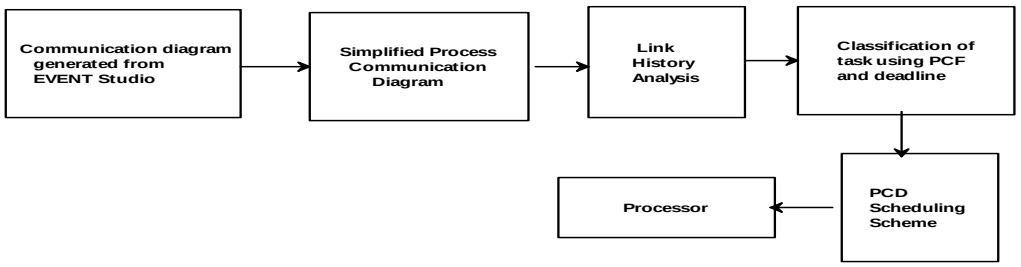


Figure 2: Block diagram of proposed framework

B. Details of the Algorithm

Here in this framework, use of design tool EVENT STUDIO 2.5 has been done to design the system and generate inter module communication diagrams and inter process communication diagrams. Advantage of EVENT STUDIO 2.5 is that it itself verifies the design and detects the resource leaks in the system. It has provision of taking review of the design i.e. it checks whether the proper resources are allocated, timers are started etc, and then only it generates design documents like sequence diagram, message interaction diagrams, module interaction diagrams etc. Secondly after getting process interaction collaboration diagram, we derive process flow diagram and this is used to predict the performability of the system.Fig.3 shows process interaction diagram generated by the EVENT STUDIO and fig.4 shows simplified process interaction diagram.

A total module is composed of number of processes and can be represented by a directed cyclic graph $G= (M, E)$ where M is set of nodes $(m \in M)$ and E is the set of edges $((e \in E)$.A node in the process interaction diagram represents a process which is a set of instructions. The edges in cyclic graph correspond to the communication messages and precedence constraints among the nodes.
Number m_{ij} associated with each node represents messages incoming to that node and outgoing from that node .Here the subscript ij indicates that the directed edges emerges from the source node n_i and incident on the destination node n_j. The source node is called parent node and destination node is called child node. In a process interaction diagram, a node which does not have any parent is called entry node while node which does not have any child is called exit node. A node can not start execution before it gathers all of the messages from its parent node.

If any message is lost in the process interaction diagram, then it is assumed that process's performance is reduced.

In a given process structure, many of processes can perform their task at different levels. In such cases, the system failure can lead to decreased ability to perform the given task, but not to complete failure. This is also called the accomplishment level of process .When a system and its components can have an arbitrary finite number of different states (task's accomplishment level) the system is termed as multi state system. The physical characteristics of the performance depend on the physical nature of system outcome.

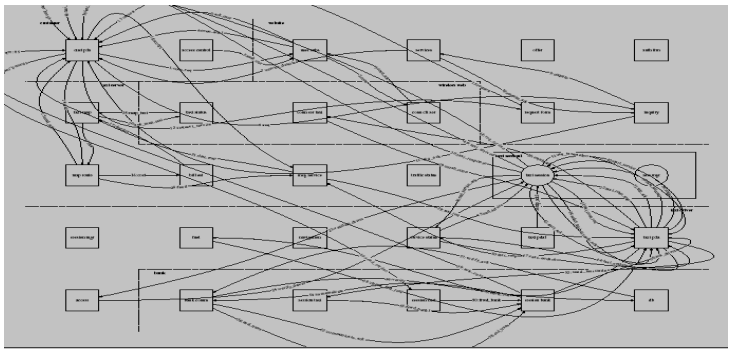


Figure 3. Process interaction diagram generated by EVENTSTUDIO.

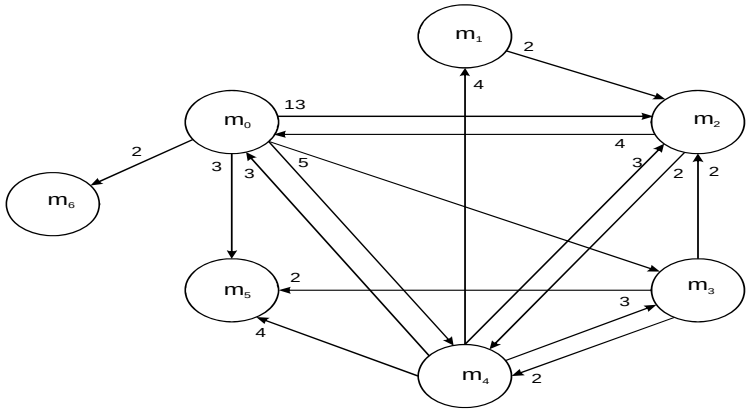


Figure 4**:** Simplified process network.

Therefore it is important to measure performance rates of system components by their contribution into the entire MSS output performance.

Notations used

G_k Output performance rate of MSS at state *k*

p_k Probability that a system is at state *k*

W_m m_{th} Possible of system demand

E_G Expected MSS performance.

E_U Expected MSS unsupplied demand

D Relative deadline

C_i Computation time

P_i Period of invocation

U_i CPU Utilization

L_{in} Total number of incoming links to a process

L_i Total links associated with that process

C Number of classes

k Total number of processes in each class

T_k^c k^{th} process in class-C

Path Predecessor paths for a process

T_p^{pr} Total number of predecessors in a path

M_p^{pr} Mandatory portion of p^{th} predecessor process

O_p^{pr} Optional portion of p^{th} predecessor process

L_C Execution time for path-C

L_{path} Execution time for precedence path

L_{total} Total execution time

Consider a process network consisting of *n* processes. It is supposed that any process unit can have k states from complete failure up to perfect functioning. The entire system has *k* different states. MSS state of instance *t* as *Y (t) ∈ {1, 2…..K}.* The performance rate G_k is associated with each state *k ∈ {1, 2….K}.*

Three most commonly used MSS reliability measures are-

1. MSS availability.
2. MSS expected performance.
3. MSS unsupplied demand (W_m).

The MSS behavior is characterized by its evaluation in the space of states. To characterize numerically this evaluation process; one has to determine MSS reliability indices.

The value of MSS expected performance could be determined as

$$E_G = \sum_{k=0}^{k} p_k G_k \qquad (1)$$

One can note that the expected MSS performance does not depend on demand W. Now from fig.4

we can calculate p_k as

$$p_k = \frac{L_{in}}{L_t} \qquad (2)$$

However here we are considering dependability of processes. Therefore we are calculating conditional probability by the following equation

$$p(m_1 \mid m_0) = \frac{n(m_1 \cap m_0)}{n(m_0)} \qquad (3)$$

If one process precedes another, then equation (3) is true. However, there are some processes in process network which are dependent on two or more processes. In the above example, m_4 is dependent on m_0 , m_3 and m_2 in that case Bay's theorem is used to evaluate conditional probability as shown in equation (4).

$$p(m_4) = p(m_0)p(m_4 \mid m_0) + p(m_3)p(m_4 \mid m_3) + p(m_2)p(m_4 \mid m_2) \qquad (4)$$

There are various possible relative accomplishment levels that characterize the performance of each process in the process network. That depends on how many incoming links are associated with that process. Various values of G_k are assumed depending upon $G_k L_{in}$

Substituting values of p_k and G_k in equation (1), various MSS performance contribution factor E_G can be calculated.

Scheduling policy for soft real time system is based on two parameters as shown in Table 1 and for hard real time system is as shown in Table 2.It is quite possible that values calculated will not be precise coming in one category.

C. For classification of processes

P be the set of processes.
C be the set of classes.
p_c be the PCF of each process.
D be the relative deadline of each process.
n be the total number of processes.
P= {p_1, p_2, p_3, p_4,…, p_n}
C= {c_1, c_2, c_3, c_4}

$$p_k \Rightarrow \{p_k \in c_1 \,|\, p_k\, c_P \wedge c_1 c_C,$$
$$P_c = High \wedge D_k = Low, n >= k > 0\} \quad (5)$$

else

$$p_k \Rightarrow \{p_k \in c_2 \,|\, p_k\, c_P \wedge c_2 c_c,$$
$$P_c = High \wedge D_k = High, n >= k > 0\} \quad (6)$$

else

$$p_k \Rightarrow \{p_k \in c_3 \,|\, p_k\, c_P \wedge c_3 c_C,$$
$$P_c = Low \wedge D_k = Low, n >= k > 0\} \quad (7)$$

else

$$p_k \Rightarrow \{p_k \in c_4 \,|\, p_k\, c_P \wedge c_4 c_C,$$
$$P_c = Low \wedge D_k = High, n >= k > 0\} \quad (8)$$

D: for allocation of process using PCD scheduling algorithm:

$$E_x = \{M,O\}\ p_1 \Rightarrow \{p_1.M = 1 \,|\, p_1.M = 0 \wedge p_1 cP, k >= 1 > 0\} \quad (9)$$

$$p_m \Rightarrow \{p_m.M = 1 \wedge p_{m.}O = 1 \,|\, p_1.M = 1,$$
$$\text{Where } k >= 1 > 0 \wedge n >= m > 0\} \quad (10)$$

Table 1.Scheduling Policy for Soft Real Time System

Priority Levels	**PCF(** E_G)	**Deadline(D)**
Class-I	High	Low
Class-II	High	High
Class-III	Low	Low
Class-IV	Low	High

Table 2. Scheduling Policy for Hard Real Time System

Priority Levels	**PCF(** E_G)	**Deadline(D)**
Class-I	High	Low
Class-II	Low	Low
Class-III	High	High
Class-IV	Low	High

Therefore frequency distribution can be used i.e. calculated values can be distributed in classes or categories to determine the number of individuals belonging to each class called class frequency.

Class 1 (PCF: 80 %-100%): $\{D_1, D_2, D_3, ...Dn\}$

Class 2 (PCF: 60 %-80%): $\{D_1, D_2, D_3, ...Dn\}$

Class 3 (PCF: 40 %-60%): $\{D_1, D_2, D_3, ...Dn\}$

Class 4 (PCF: below 40 %) :{ $\{D_1, D_2, D_3, ...Dn\}$

Assumptions

- Tasks are divided into mandatory and optional portion.
- Data required for transmission to successor are processed by mandatory portion.
- For scheduling dependent tasks, if predecessors are from class-I, then its mandatory as well as optional portion is executed. If predecessors are from other than class-I, tasks are scheduled only for mandatory portion only.
- The semantics assumed is that one instance of all tasks should be executed every period. This scheduler is non pre-emptive type.
- Further it is assumed that, total computation of task required that it should get all messages properly. If any of message from predecessor fails, its accomplishment level reduces accordingly the period of task.

E. Suggested scheduling policy

```
For index1=class1 to class4
 index2=process1 to MAXprocessINDEX1   num=calculate_preceded_process(index2);
                    for index3=0 to num                    if(preceded[index3].mandatory=0      then
                              execute(preceded[index3].mandatory);
          setflag(preceded[index3].mandatory);
                    endif
          endfor
                    if(process[index2].mandatory=0 then
        execute(process[index2].mandatory);  setflag(process[index2].mandatory);
                    endif
                    if(process[index2].optional=0 then
          execute(process[index2].optional);
                    setflag(process[index2].optional);
                    endif
          endfor
endfor
```

F: Evaluation of Algorithm

This algorithm as well as LDF is simulated in VB and MS Access. For analyzing our algorithm, we have done many case studies. The process structure in fig. 5 shows the dependent processes.

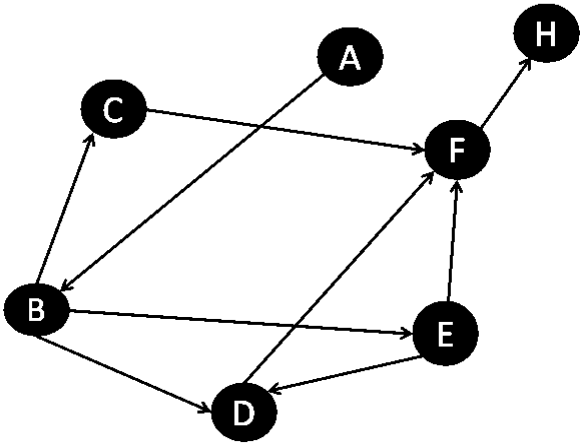


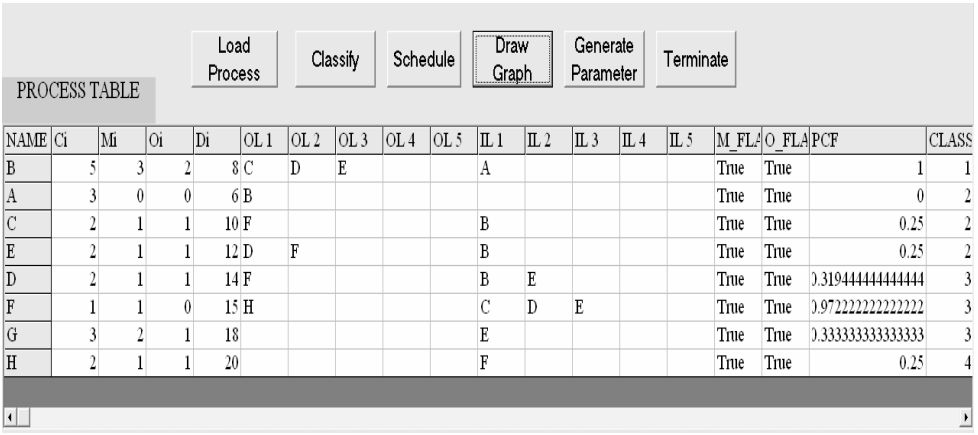
Figure 5: Process network

We stored the process information in database in MS-Access. On clicking the Classify button on main form, PCF of each process is calculated and processes are classified into priority classes based on deadline and PCF as shown in fig. 6.

According to priority classes, the processes are executed and result is as shown in fig.7.

The graph of execution of processes according to PCD is drawn as shown in figure 8.

The same task set is applied to LDF which is one of the existing schedulers. Fig.9 shows the number of missing deadlines by LDF. Scheduling graph of LDF is shown in fig.10.



NAME	Ci	Mi	Oi	Di	OL 1	OL 2	OL 3	OL 4	OL 5	IL 1	IL 2	IL 3	IL 4	IL 5	M_FLA	O_FLA	PCF	CLASS
B	5	3	2	8	C	D	E			A					True	True	1	1
A	3	0	0	6	B										True	True	0	2
C	2	1	1	10	F					B					True	True	0.25	2
E	2	1	1	12	D	F				B					True	True	0.25	2
D	2	1	1	14	F					B	E				True	True	0.319444444444444	3
F	1	1	0	15	H					C	D	E			True	True	0.972222222222222	3
G	3	2	1	18						E					True	True	0.333333333333333	3
H	2	1	1	20						F					True	True	0.25	4

Figure 6: Processes classified according to PCF and Deadline.

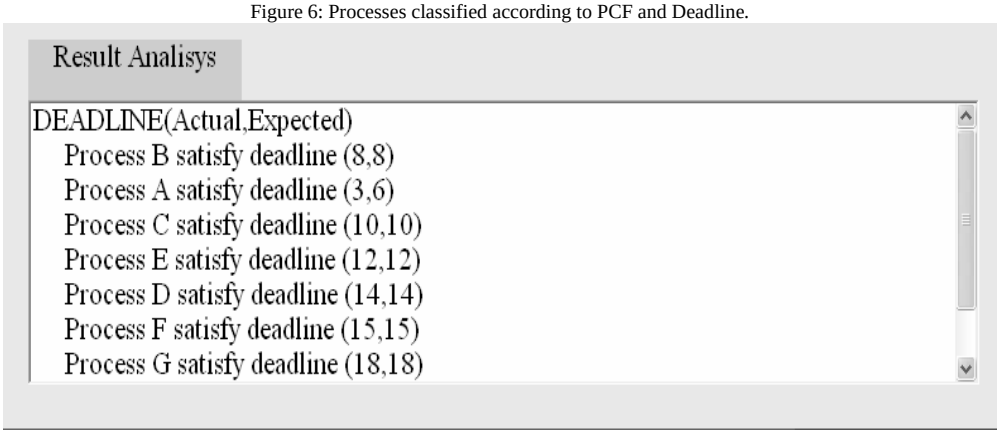


Figure 7: Result of PCD scheduler

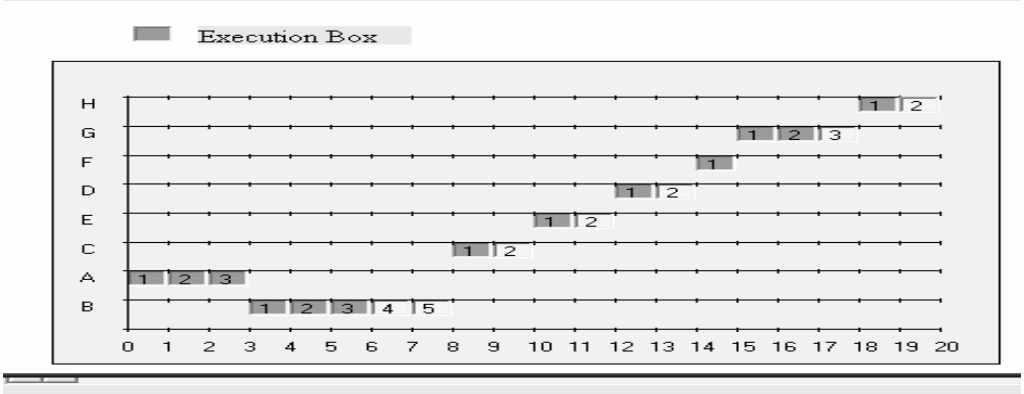


Figure 8: Graph drawn after scheduling Processes by PCD

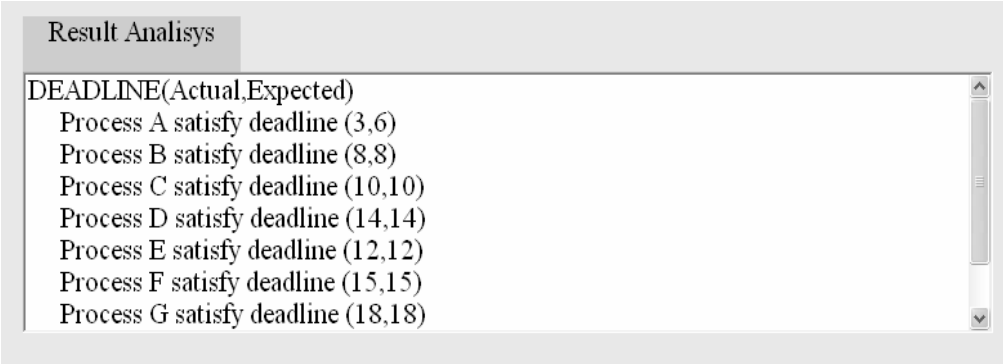


Figure 9: Result of LDF scheduler

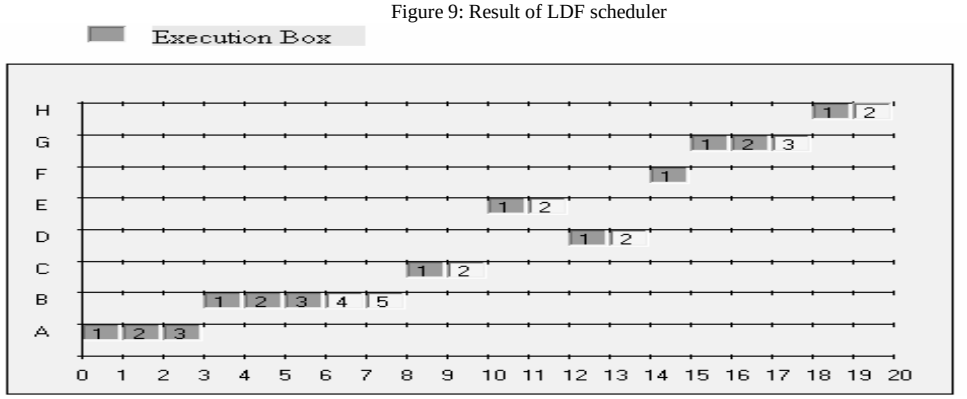


Figure 10: Graph drawn after scheduling processes by LDF

From above example it is clear that LDF considers only deadline but our designed scheduler considers both deadline and PCF. So as processes having higher PCF are executed first, the performance of the system is increased.

The other most important advantage of PCD over LDF is that it can schedule the processes having cycle in the process structure diagram. LDF can not schedule such processes. We have carried 4 such case studies and summary of these case studies is as shown in Figure 11 and 12

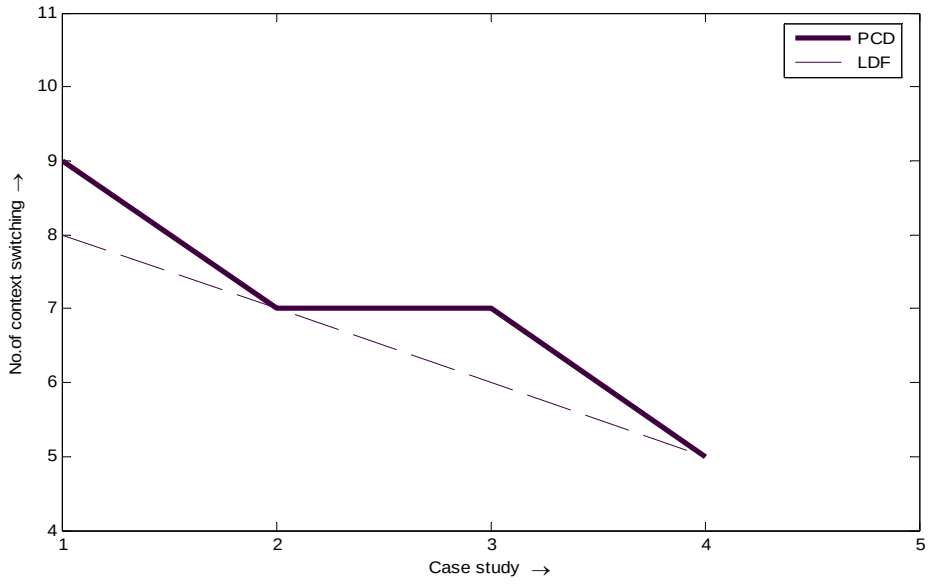


Figure 11: Context switching of PCD with LDF

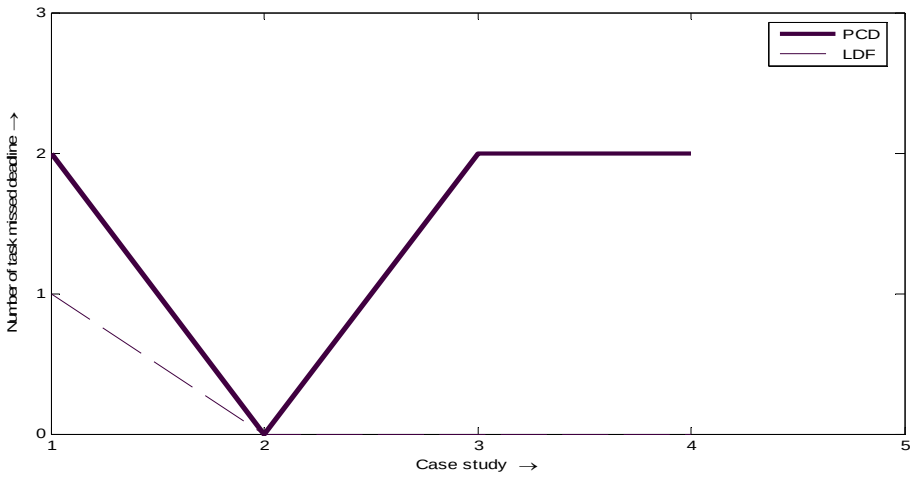


Figure 12: Missing of deadlines of PCD with LDF

From figure 11 and 12, although context switching in PCD is higher but it is trying to schedule processes in cyclic graph. Similarly deadline missing in PCD is slightly higher than PCD.

G. Future work

The proposed algorithm is simulated for uniprocessor and scheduling proposed is non preemptive. However non preemptive scheduling is usually considered inferior to preemptive scheduling, because the non preemptive block would lead to poor task responsiveness [21].However Baruah and Chakraborty [22] addressed schedulability analysis for non preemptive recurring tasks, which is the general form of non preemptive tasks and showed that the non preemptive schedulability analysis problem can be reduced to a polynomial number of preemptive schedulability analysis problem. If we will use the same for multi processor or distributed architecture. The context switching and deadline missing ratio will reduce considerably. As this analysis is at design phase ,one can adjust deadlines of predecessor tasks accordingly so as to get better schedulability and context switching.

VI. CONCLUSION

Relative contribution of each process in process structure needs to be evaluated prior to schedule the process to processor. In this paper it is suggested to have pre analysis of design to find out the same. The paper proposes a novel idea to find contribution of each process in process structure using multi state system analysis concept. Although LDF is optimal scheduling algorithm, it does not suit for process structures which are cyclic. We have tried to find out solution for cyclic process structure. This particular framework also considers performability aspect of real time scheduler. It is observed that results are encouraging.

REFERENCES

[1] C. L. Liu and J. W. Layland “Scheduling algorithms for multiprogramming in a hard-real-time environment,”Journal of the ACM, 20(1), pp.47-61January 1973.
[2] J.Sitakis, “Modeling real time systems-challenges and work directions,” proceedings of the int. Conference on Embedded Software(EMSOFT 2001), Springer, LNOS 2211, 2001.
[3] K.Alstisen, G.Goessler and J.Sitakis, “Scheduler modeling based on the controller synthesis paradigm,” Journal of real time system, special issue on control approaches to Real Time computing.23.pp.55-84,2002.
[4] P.Binns and S.Vesta , “Formalizing software architecture for embedded system ,” proceeding of the first Int. Conference on embedded software(EMSOFT 2001), Tohae city, Springer, 2001
[5] V.Bertin, E.Closse, M.Poize atel, “Taxys = Esterel t Kronos, a tool for verifying real time properties of embedded system ,”Proceedings the conference on Decision and control ,Dec-2001.
[6] J.Sitakis, S.Tripakis, S.Yovine, “Building models of real time system from application software, ” Proceeding of the IEEE ,special issues on modeling and design of embedded ,pp.100-111,Jan. 2003.
[7] E.G. Coffman, “Computer and Job-Shop Scheduling Theory,” New York: John Wiley & Sons, 1976.
[8] M.R. Garey and D.S. Johnson, “Computers and Intractability: A Guide to the Theory of NP-Completeness, ” W.H. Freeman Co., 1979.
[9] J.K. Lenstra and A.H.G.R. Kan, “Complexity of Scheduling Under Precedence Constraints,” Operations Research, vol. 26, no. 1, pp. 23- 35, Jan. 1978.

[10] E.L. Lawler, “Deterministic and Stochastic Scheduling,: Recent Developments in Deterministic Sequencing and Scheduling: A Survey,” pp. 35-74. The Netherlands: Reidel, Dordrecht, 1982.
[11] H. Kasahara and S. Narita, “Practical Multiprocessor Scheduling Algorithms for Efficient Parallel Processing, ” IEEE Trans. Computers, vol. 33, no. 11, pp. 1,023-1,029, Nov. 1984.
[12] W.W. Chu and K. Leung, “Module Replication and Assignment for Real-Time Distributed Processing Systems,” Proc. IEEE, vol. 75, no. 5, pp. 547-562, May 1987.
[13] C.C. Shen and W.H. Tsai, “A Graph Matching Approach to Optimal Task Assignment in Distributed Computing Systems Using a Minimax Criterion, ” IEEE Trans. Computers, vol. 34, no. 3, pp. 197- 203, Mar. 1985.
[14] P.Y.R. Ma et al., “A Task Allocation Model for Distributed Computing Systems,” IEEE Trans. Computers, vol. 31, no. 1, pp. 41-47, Jan. 1982.
[15] J.B. Sinclair, “Efficient Computation of Optimal Assignments for Distributed Tasks," J. Parallel and Distributed Computing, vol. 4, pp. 342-362, 1987.
[16] W.H. Kohler and K. Steiglitz, “Computer and Job-Shop Scheduling Theory: Enumerative and Iterative Computational Approach," pp. 229-287. John Wiley & Sons, 1976.
[17] M. Alfano. A. Di-Stefano, L. Lo-Bello, O. Mirabella, and J.H. Stewman, “An Expert System for Planning Real-Time Distributed Task Allocation,” Proc. Florida AI Research Symp., Key West, Fla.,May 1996.
[18] P. Altenbernd, C. Ditze, P. Laplante, and W. Halang, “Allocation of Periodic Real-Time Tasks,” Proc. 20th IFAC/IFIP Workshop, Fort Lauderdale, Fla., Nov. 1995.
[19] Hong Pham “Handbook of reliability engineering” Springer publication,pp.60-68,2003.
[20] E.L.Lawler “Optimal sequencing of a single machine subject to precedence constraints” JSTOR management science, vol.19 No.5 Jan , pp.544-5461973.
[21] Nan Guan,Wang Yi,Qingxu Deng,Zonghua Gu, Ge Yu" Schedulability analysis for non-preemptive fixed priority multiprocessor scheduling," Journal of System architecture,pp.1-10,2010.
[22] S.K.Baruah,S Chakraborty "Schedulability analysis of non-preemptive recurring real time tasks" In 14th international workshop on Parallel and Distributed Real Time Systems,2006.

Radhakrishna Naik received his B.E(1997)from Govt.COE Auarangabad. and M.Tech from CEDTI Aurangabad at Dr.BAM University Aurangabad in 2004.Currently he is working as Associate Professor in IT Dept. SRES COE Kopargaon and he is a PhD student at University of SRTM University Nanded in the SGGS COE Nanded. Under the supervision of Dr.R.R.Manthalkar. His research interests include real time systems and software reliability.

Dr. R.R. Manthalkar received his B.E (1988) and M.E.(Electronics) (1994) from SGGS Institute of Engg & Technology, Nanded, India.He was Rank I for BE(Electronics) in the University. He completed Ph.D. from IIT Kharagpur (2003). He has published many research papers in Journals (9) and Conferences (25). His research interests are Texture Classification and Pattern Recognition,Real time systems. Currently he is Professor and Head of Electronics Dept. in SGGS Institute of Engg & Technology, Nanded, M.S.,India