PATTSAS: Periodic and Aperiodic Real-Time Task Scheduling Algorithms Simulator

Radhakrishna Naik
Dept. of Comp. Sci. & Engg
MIT, Aurangabad, India
naikradhakrishna@gmail.com

Mukta Dhopeshwarkar
Dept. of IT&CS
Dr. B.A.M. University
muktad@rediffmail.com

R.R.Manthalkar
SGGSCOE, Nanded, India
rrmanthalkar@yahoo.com

K.Vengatesan.
Dept. of Computer Engineering
SRESCE, Kopargaon,
K.Vengatesan.

Abstract— This paper describes a framework called PATSAS which provide tool to check various scheduling algorithms simulation to predict the behavior of the system and also checks if the real time task meets the temporal constraint. This framework is based on different scheduling algorithms provided by different authors for scheduling of periodic and aperiodic tasks. PATSAS is using the periodic and aperiodic tasks as primary input and also the precedence relationship among the tasks and gives the total scheduling of that tasks with CPU utilization, tasks missed deadline, context switches occurred during scheduling, number of preemptions performed. It provide an feasible simulation engine which allows the system designer to predict the system future and system behavior along with system resource usage.

Keyword– Rate Monotonic Scheduling (RMS), Earliest Deadline First (EDF), Deferrable Server (DS), Priority Exchange (PE), Instantaneous Utilization Factor (IUF).

I. INTRODUCTION

This paper presents a real time simulating framework PATSAS. PATSAS provides facility to study and analyze temporal behavior of real time tasks. Most of the tasks need to meet some essential constraints such as response time, deadline, CPU utilization, precedence of tasks. Designer has to select temporal parameters by trial and error method. It takes lot time and efforts. Since from 1980 onwards there are many tools are available but none of the tool has got wide coverage of algorithms to be simulate. PATSAS satisfies this need. It can be used to simulate an around 12 popular real time scheduling algorithm which deals with periodic, aperiodic and precedence constrained tasks. Here designer can select all scheduling attributes verify their results.

II. RELATED WORK

Real time scheduling theory helps the system designer to analyze the timing behavior of a set of tasks with scheduling algorithms or with algebraic methods usually called feasibility tests. Scheduling algorithms allow the designer to compute scheduling simulations of the architecture to analyze.

There are various such tools are available however none of them can be used as generalized tool.
accepts input as timing profiles. These timing profiles are analyzed for various possible problems. It suggests changing task period, change in execution time, it also suggest possibility of static and dynamic scheduling.

The MAST [15] Ada frameworks also provides some basic feasibility tests and simulation services for fixed priority schedulers. The framework is portable enough and open source. Unfortunately, if simulations with a new scheduler or new task activation pattern must be done, the simulator has to be modified. Defining specific schedulers may be difficult to do for students.

Realtss [16], an open source real time scheduling simulator. Realtss is aimed to be used to test and evaluate existing and new real-time scheduling algorithms. It is composed of a graphical front-end to set task's parameters and to view simulation results. Its modular design allows the integration of additional scheduling policies seamlessly. New scheduling algorithms can be added as modules written in TCL, C or C++ and are not limited by a particular simulation language. It can be executed in many operating systems, such as Linux or Windows.

Finally, CHEDDAR is a free real time scheduling tool [17]. CHEDDAR is an Ada framework which provides tools to check if a real time application meets its temporal constraints. This framework is based on the real time scheduling theory. With cheddar, an application is defined by a set of processors, tasks, buffers, shared resources and message. It also provides feasibility tests in the case of monoprocessor, multiprocessor and distributed system.

III. PATSAS

This paper presents a real time scheduling framework called PATSAS, which stands for Periodic and Aperiodic Task Scheduling Algorithm Simulator.

Since, no one tool providing the facility of algorithms like precedence constrained task scheduling, Aperiodic tasks scheduling algorithms, Algorithms which support scheduling to sustain in transient faults such as IUF and MIUF.

It provides tools to check if a real time application meets its temporal constraints such as response time, number of preemptions, context switching and CPU utilization. The framework is based on real time scheduling theory and is mostly written for educational purposes. With PATSAS an application defined by tasks and its parameters.

PATSAS provides a flexible simulation engine which allows designer to describe and run simulations of specific systems.

The proposed framework measures the following parameters of an scheduling algorithm:

- Utilization of processor
- Response time
- Number of context switch
- Number of pre-emption
- Task missing deadline
- Gantt chart of each scheduler

The framework supports for following algorithms:

- RMS (Rate Monotonic Scheduling) of dependent task
- LDF (Latest Deadline First) of dependent task
- EDF (Earliest Deadline First) of dependent task
- DMS (Deadline Monotonic Scheduling) of dependent task
- RMS of independent task
- EDF of independent task
- Instantaneous Utilization First
- Modified Instantaneous Utilization First
- Polling Server
- Deferrable Server
- Sporadic Server

The System mainly takes periodic and aperiodic tasks as an input with information like release time, computation time, period and deadline and also some algorithm specific information like mandatory and optional part of task needed for IUF, Precedence relationship among tasks as shown in Fig.3 for simulation of dependent task, server capacity and server period for simulation of aperiodic task server.

Fig. 1 shows the main screen of PATSAS tool showing the supported algorithms lists. User need to choose one algorithm from it.

Fig. 2 shows the complete task information input structure of the PATSAS.

Figure 1: PATSAS screen showing supported algorithms.
In this section, we are evaluating behavior of different algorithms depending on input task set.

a) Performance evaluation of dependent tasks

Table I. Input task set for dependent tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Ri</th>
<th>Ci</th>
<th>Di</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>0</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>T2</td>
<td>0</td>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>T3</td>
<td>0</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>T4</td>
<td>0</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>T5</td>
<td>0</td>
<td>2</td>
<td>9</td>
</tr>
</tbody>
</table>

For evaluating the performance of dependent task the information of task set applied to simulator is given in table 1, and the precedence relationship among those tasks are shown in fig. 3.

Following fig. 4 shows rate monotonic scheduling of dependent tasks provided in Table 1.

Fig. 5 Shows simulation with deadline monotonic scheduling.

Fig. 6: Latest Deadline First scheduling of dependent tasks simulation
Fig. 6 shows the simulation with LDF scheduling. Fig. 7 shows simulation with EDF. Each figure is showing complete scheduling of each task along with information like tasks missed deadline, number of context switches occurred etc. and also CPU utilization.

Table II gives summarized information about the simulation done by various algorithms and results obtained.

### Table II: Result analysis of RMS, EDF, DMS, LDF

<table>
<thead>
<tr>
<th>ALGORITHM</th>
<th>No. Of context switching</th>
<th>CPU Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMS</td>
<td>4</td>
<td>0.99</td>
</tr>
<tr>
<td>EDF</td>
<td>4</td>
<td>0.99</td>
</tr>
<tr>
<td>DMS</td>
<td>4</td>
<td>0.99</td>
</tr>
<tr>
<td>LDF</td>
<td>4</td>
<td>0.99</td>
</tr>
</tbody>
</table>

### Performance Evaluation of independent tasks

Table III: Input Task set for independent task

<table>
<thead>
<tr>
<th>Task</th>
<th>Arrival Time</th>
<th>Execution Time</th>
<th>Period</th>
<th>Deadline</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>0</td>
<td>25</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>T2</td>
<td>0</td>
<td>10</td>
<td>65</td>
<td>20</td>
</tr>
<tr>
<td>T3</td>
<td>0</td>
<td>25</td>
<td>125</td>
<td>50</td>
</tr>
</tbody>
</table>

Table 3 shows the task set applied for simulation of independent tasks. Following subsequent figures shows simulation done by the PATSAS. The fig. 8 shows RMS simulation, Fig. 9 shows EDF simulation and fig. 10 shows IUF simulation.
Following Table IV shows summarized results about simulation.

Table IV: Analysis of RMS, EDF and IUF

<table>
<thead>
<tr>
<th></th>
<th>RMS</th>
<th>EDF</th>
<th>IUF</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU utilization</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
</tr>
<tr>
<td>No. of Context switching</td>
<td>172</td>
<td>181</td>
<td>195</td>
</tr>
<tr>
<td>No. of preemption</td>
<td>33</td>
<td>43</td>
<td>195</td>
</tr>
<tr>
<td>No. of deadline missed</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

c) Performance evaluation of Servers

Table V: Input task set aperiodic server simulation

<table>
<thead>
<tr>
<th>Task</th>
<th>Arrival time</th>
<th>Execution time</th>
<th>Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0</td>
<td>5</td>
<td>55</td>
</tr>
<tr>
<td>P2</td>
<td>0</td>
<td>4</td>
<td>92</td>
</tr>
<tr>
<td>P3</td>
<td>0</td>
<td>9</td>
<td>165</td>
</tr>
<tr>
<td>A1</td>
<td>2</td>
<td>22</td>
<td>-</td>
</tr>
</tbody>
</table>

Fig. 11 shows simulation of above task set done by polling server (PS). Fig. 12 shows simulation done by Defferable Server (DS). Fig. 13 shows the simulation done by Sporadic server (SS).

Following table VI shows the result analysis of performance of PS, DS, SS.

Table VI: Analysis of PS, DS and SS

<table>
<thead>
<tr>
<th></th>
<th>Polling server</th>
<th>Deferrable server</th>
<th>Sporadic server</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU utilization</td>
<td>0.67</td>
<td>0.67</td>
<td>0.67</td>
</tr>
<tr>
<td>Response time</td>
<td>112</td>
<td>58</td>
<td>60</td>
</tr>
</tbody>
</table>

V. CONCLUSION

We have done analysis of already available scheduling algorithm simulators and tried to overcome the drawback of previous simulators like cheddar by providing more friendly user interface(GUI). The proposed simulator simulates different types of scheduling algorithms that schedules dependent, independent as well as aperiodic tasks. The proposed system also supports precedence.
constraint algorithm as RMS with precedence constraint, EDF with precedence constraint, DMS with precedence constraint, LDF with precedence constraint, also algorithms like RMS and EDF for independent task. For scheduling of aperiodic task framework provides server as polling server, deferrable server and sporadic server.

Also framework calculates parameters such as number of context switching, number of preemption, CPU utilization.

VI. REFERENCES


