CHAPTER 5

SCHEDULING BASED ON FUZZIFIED NON-UNIFORM LAXITY

Scheduling analysis can be carried out as long as task parameters like execution time, deadline, etc are precisely available. However in many real time situations, these parameters are not precisely available, thus making scheduling analysis difficult. In such situations, the task parameters are specified as linguistic variables and the same are handled based on a set of rules using a fuzzy inference engine. This chapter proposes a scheduling algorithm that uses a fuzzified non-uniform laxity based approach.

The chapter is organised as follows. Section 5.1 explains the model of the fuzzy inference engine based scheduler. Section 5.2 discusses the nature and type of membership functions used and the basis on which the rule base is formulated for the fuzzy inference engine. Section 5.3 explains the salient points of the algorithm. Section 5.4 explains the simulation environment and discusses the results in detail. Section 5.5 concludes the chapter.

5.1 MODEL OF FUZZY INference ENGINE BASED SCHEDULER

Existing fuzzy based approaches like FPMLF, FGEDF, etc (Sabeghi et al 2006) use different scheduling parameters like execution time, laxity, deadline, etc. as linguistic inputs and obtain a revised priority for execution. The choice of the variable used as an input to the fuzzy inference engine plays a key role in the effectiveness of the scheduling analysis.
this work, non-uniform laxity and priority are used as inputs to the fuzzy inference engine. The output of the fuzzy inference engine is execution eligibility. The non-uniform laxity indicates the exact slack availability before the scheduling is carried out. A Mamdani's engine is used which accepts inputs as fuzzy sets and does not require the inputs to be linear. The block diagram of the fuzzy inference scheduler is shown in Figure 5.1

![Block diagram of fuzzy inference engine and scheduler](image)

**Figure 5.1 Block diagram of fuzzy inference engine and scheduler**

The scheduling is done in two phases:

- Fuzzification phase
- Scheduling phase

### 5.1.1 Fuzzification Phase

The fuzzy inference engine analyses the inputs based on a set of rules and gives the order of execution, which is the execution eligibility. Based on the execution eligibility, the tasks are dispatched to the scheduler. While deciding upon the execution order of tasks, the slack position is also taken into account, by the fuzzy inference engine.
5.1.2 Scheduling Phase

Based on the execution order decided in the fuzzification phase, the scheduling process is commenced. Tasks are scheduled based on the minimum resource requirement, storing the remaining quantum of resource as slack. If a task runs out of resource or the same misses a deadline, resource is borrowed from the cumulated slack after computation of the exact requirement of resource by the task. After obtaining the resource, a relative deadline, is set and the task is scheduled.

The next section explains the types of membership functions chosen and the basis of formulation for the rule base.

5.2 MEMBERSHIP FUNCTIONS AND RULE BASE

The non-uniform laxity is computed from the deadline of tasks. Both the priority and non-uniform laxity take four different values as Very Low (VL), Low (L), High (H) and Very High (VH). The normal range is between Low and High. The membership functions are triangular. Priority of tasks takes values VL, L, H and VH for ranges (0-3), (2-5), (4-7) and (6-10) respectively. Deadlines of tasks take values VL, L, H and VH for ranges (0-25), (15-38), (32-54) and (46-70) respectively. Non-uniform laxity takes values VL, L, H and VH for ranges (0-38), (27-63), (55-90) and (81-117) respectively. The membership functions with respective ranges for the linguistic variables are shown in Figures 5.2, 5.3 and 5.4.
The fuzzification is carried out based on a set of rules. A typical rule like, ‘If (priority is very low) and (non-uniform laxity is very low), then (execution eligibility is low)’, indicates the fact that a task with very low priority and very low non-uniform laxity is of least importance. These tasks are scheduled at a later point of time, depending on the slack availability.
Other rules are formulated in a similar manner. The sixteen rules based on which the rule base is formulated is given below:

- If (priority is very low) and (non-uniform laxity is very low) then (execution eligibility is low).
- If (priority is very low) and (non-uniform laxity is low) then (execution eligibility is low).
- If (priority is very low) and (non-uniform laxity is high) then (execution eligibility is very low).
- If (priority is very low) and (non-uniform laxity is very high) then (execution eligibility is very low).
- If (priority is low) and (non-uniform laxity is very low) then (execution eligibility is high).
- If (priority is low) and (non-uniform laxity is low) then (execution eligibility is low).
- If (priority is low) and (non-uniform laxity is high) then (execution eligibility is low).
- If (priority is low) and (non-uniform laxity is very high) then (execution eligibility is low).
- If (priority is high) and (non-uniform laxity is very low) then (execution eligibility is very high).
- If (priority is high) and (non-uniform laxity is low) then (execution eligibility is high).
- If (priority is high) and (non-uniform laxity is high) then (execution eligibility is low).
- If (priority is high) and (non-uniform laxity is very high) then (execution eligibility is low).
- If (priority is very high) and (non-uniform laxity is very low) then (execution eligibility is very high).
• If (priority is very high) and (non-uniform laxity is low) then (execution eligibility is very high).

• If (priority is very high) and (non-uniform laxity is high) then (execution eligibility is high).

• If (priority is very high) and (non-uniform laxity is very high) then (execution eligibility is low).

The next section explains the algorithm.

5.3 ALGORITHM

• Feed priority and non-uniform laxity of tasks to the fuzzy inference engine

• Get the execution eligibility for each task based on the rules set

• Based on this execution eligibility try to schedule the task with minimum execution time

• For each core do

  • For each task do

    • If the task can be scheduled with minimum execution time before deadline

      • Schedule the task with minimum execution time

      • Compute the task utilisation

      • If (Σ task utilisations <= 1)

        • Schedule task on same core.

      Else

        • Schedule task on next core

    Else

      • Compute the required time needed, remaining slack time
• Update the slack with reference to the used slack time by the task
  • Schedule the task with slack time
  fi
  od
  od
• End the procedure.
The next section presents the simulation results and discusses them.

5.4 SIMULATION RESULTS

The fuzzified non-uniform laxity approach has been compared against the existing fuzzy based approaches. The algorithm has been tested varying the number of cores up to 100 and tasks up to 150. Evaluation is done with respect to resource utilization and task schedulability.

5.4.1 Resource Utilization

The fuzzified non-uniform laxity based approach has been compared against the existing fuzzy based approaches, namely, FGMLF and FPMLF (Sabheghi et al 2005). The results are tabulated in Table 5.1. For example, when 6 tasks are scheduled on 4 cores, the fuzzy based non-uniform laxity approach utilizes 74.0 units, compared to FGMLF and FPMLF which utilize 63.0 and 68.0 units respectively. The fuzzified non-uniform laxity based approach improves resource utilization, compared to FGMLF and FPMLF, by 30% and 13% respectively. The results are illustrated in Figure 5.5.
Table 5.1 Improvement in resource utilization- Fuzzified non-uniform laxity approach Vs existing fuzzy based approaches-using SESC

<table>
<thead>
<tr>
<th>Number of cores</th>
<th>Number of tasks</th>
<th>Slack resource utilization</th>
<th>Improvement for FNLAX over FGMLF (%)</th>
<th>Improvement for FNLAX over FPMLF (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FGMLF</td>
<td>FPMLF</td>
<td>FNLAX</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>66</td>
<td>68</td>
<td>74</td>
<td>17.5</td>
</tr>
<tr>
<td>8</td>
<td>58</td>
<td>63</td>
<td>69</td>
<td>19.0</td>
</tr>
<tr>
<td>12</td>
<td>49</td>
<td>55</td>
<td>61</td>
<td>24.5</td>
</tr>
<tr>
<td>16</td>
<td>44</td>
<td>51</td>
<td>57</td>
<td>29.5</td>
</tr>
<tr>
<td>32</td>
<td>41</td>
<td>47</td>
<td>53</td>
<td>29.3</td>
</tr>
<tr>
<td>64</td>
<td>38</td>
<td>44</td>
<td>51</td>
<td>34.2</td>
</tr>
<tr>
<td>100</td>
<td>30</td>
<td>39</td>
<td>47</td>
<td>56.7</td>
</tr>
<tr>
<td>Average</td>
<td>46.1</td>
<td>52.4</td>
<td>58.9</td>
<td>30.1</td>
</tr>
</tbody>
</table>

Figure 5.5 Improvement in resource utilization- Fuzzified non-uniform laxity approach Vs existing fuzzy based approaches-using SESC
5.4.2 Task Schedulability

The fuzzified non-uniform laxity based approach has been compared against the existing fuzzy based approaches, namely, FGMLF and FPMLF (Sabheghi et al 2005). The results are tabulated in Table 5.2. For example, when 6 tasks are to be scheduled, the fuzzy based non-uniform laxity approach schedules 5 tasks, compared to FGMLF and FPMLF which schedules only 3 and 4 tasks respectively. The fuzzified non-uniform laxity based approach improves task schedulability, compared to FGMLF and FPMLF, by 45% and 15% respectively. The results are illustrated in Figure 5.6.

### Table 5.2 Improvement in task schedulability - Fuzzified non-uniform laxity approach Vs existing fuzzy based approaches-using SESC

<table>
<thead>
<tr>
<th>Number of tasks</th>
<th>Task schedulability</th>
<th></th>
<th></th>
<th>Improvement</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FGMLF</td>
<td>FPMLF</td>
<td>FNLAx</td>
<td>FNLAx over FGMLF (%)</td>
<td>FNLAx over FPMLF (%)</td>
</tr>
<tr>
<td>6</td>
<td>Tasks scheduled</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>66.7</td>
</tr>
<tr>
<td></td>
<td>Tasks missed</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>80.0</td>
</tr>
<tr>
<td>12</td>
<td>Tasks scheduled</td>
<td>5</td>
<td>7</td>
<td>9</td>
<td>60.0</td>
</tr>
<tr>
<td></td>
<td>Tasks missed</td>
<td>7</td>
<td>4</td>
<td>2</td>
<td>57.1</td>
</tr>
<tr>
<td>18</td>
<td>Tasks scheduled</td>
<td>10</td>
<td>14</td>
<td>16</td>
<td>24.3</td>
</tr>
<tr>
<td></td>
<td>Tasks missed</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>14.8</td>
</tr>
<tr>
<td>24</td>
<td>Tasks scheduled</td>
<td>14</td>
<td>19</td>
<td>22</td>
<td>10.5</td>
</tr>
<tr>
<td></td>
<td>Tasks missed</td>
<td>10</td>
<td>5</td>
<td>2</td>
<td>44.8</td>
</tr>
<tr>
<td>48</td>
<td>Tasks scheduled</td>
<td>37</td>
<td>42</td>
<td>46</td>
<td>44.8</td>
</tr>
<tr>
<td></td>
<td>Tasks missed</td>
<td>11</td>
<td>6</td>
<td>2</td>
<td>44.8</td>
</tr>
<tr>
<td>96</td>
<td>Tasks scheduled</td>
<td>81</td>
<td>87</td>
<td>93</td>
<td>44.8</td>
</tr>
<tr>
<td></td>
<td>Tasks missed</td>
<td>15</td>
<td>9</td>
<td>3</td>
<td>44.8</td>
</tr>
<tr>
<td>150</td>
<td>Tasks scheduled</td>
<td>133</td>
<td>142</td>
<td>147</td>
<td>44.8</td>
</tr>
<tr>
<td></td>
<td>Tasks missed</td>
<td>132</td>
<td>141</td>
<td>3</td>
<td>44.8</td>
</tr>
</tbody>
</table>

*Average*
Figure 5.6 Improvement in task schedulability - Fuzzified non-uniform laxity approach Vs existing fuzzy based approaches-using SESC

5.5 SUMMARY

This chapter has proposed the fuzzified non-uniform laxity approach, in situations where there is lack of precise knowledge of the scheduling parameters. Simulations show improvement in resource utilization and task schedulability.

The scheduling approaches discussed in chapters 3, 4 and 5 are effective as long as tasks are independent in nature. In real time situations, tasks may have dependencies. The scheduling analysis has to be modified when such task dependencies are encountered, which is addressed in the next chapter.