Combining Task-level and System-level Scheduling Modes for Mixed Criticality Systems

DS-RT 2019, Cosenza Italy, 2019-10-08

Jalil Boudjadar\textsuperscript{1}
Saravanan Ramanathan\textsuperscript{2} and Arvind Easwaran\textsuperscript{2}
Ulrik Nyman\textsuperscript{3} ulrik@cs.aau.dk

\textsuperscript{1}Aarhus University, Denmark
\textsuperscript{2}Nanyang Technological University, Singapore
\textsuperscript{3}Distributed, Embedded and Intelligent Systems, DEIS
Department of Computer Science
Aalborg University, Denmark
Combining Task-level and System-level Scheduling Modes for Mixed Criticality Systems
Setting and Purpose

Mixed-critical systems

- High criticality tasks (HC)
- Low criticality tasks (LC)
Setting and Purpose

Mixed-critical systems

- High criticality tasks (HC)
- Low criticality tasks (LC)
- HC tasks should never miss a deadline
- Dropping as few LC tasks as possible
Related Work

Two approaches

System-level mode

- Two system modes
  - Normal
  - Critical

Task-level mode
Related Work

Two approaches

System-level mode

- Two system modes
  - Normal
  - Critical
- Drop all LC tasks
- Run LC tasks in degraded mode

Task-level mode
Related Work
Two approaches

System-level mode
- Two system modes
  - Normal
  - Critical
- Drop all LC tasks
- Run LC tasks in degraded mode

Task-level mode
- Each HC task switches individually to Critical
Related Work
Two approaches

System-level mode
- Two system modes
  - Normal
  - Critical
- Drop all LC tasks
- Run LC tasks in degraded mode

Task-level mode
- Each HC task switches individually to Critical
- Other HC tasks can miss their deadline
Assumptions

- Tasks are preemptible.
- All tasks are assigned a static criticality level (LC or HC) by design, called default criticality.
- The execution of a HC task must not be discarded under any runtime circumstances.
- The runtime criticality of a LC task can never be upgraded to HC.
- LC tasks stick always to their low confidence WCET.
- There is no dependency between LC and HC tasks.
A task $\pi_i$ is given by $\langle T_i, C^l_i, C^h_i, \chi_i, \rho \rangle$ where:

- $T_i$ is the task period.
- $C^l_i \in \mathbb{R}_{\geq 0}$ and $C^h_i \in \mathbb{R}_{\geq 0}$ are the worst case execution time for low and high confidence levels respectively. We assume that $C^h_i \geq C^l_i$ for HC tasks, and $C^h_i = C^l_i$ for LC tasks.
- $\chi_i \in \{\text{LC, HC}\}$ is the default (constant) criticality of the task.
- $\rho$ is the task priority.

The task runtime mode $\Omega(\cdot)$ will be updated on the fly according to the actual task execution budget.
### Task types

#### Types of tasks

<table>
<thead>
<tr>
<th></th>
<th>HC</th>
<th>LC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>High criticality</td>
<td>Low criticality</td>
</tr>
<tr>
<td>Modes</td>
<td>HI and LO</td>
<td>no modes</td>
</tr>
<tr>
<td>WCET</td>
<td>Varies $C_i^h \geq C_i$</td>
<td>Static $C_i^h = C_i$</td>
</tr>
</tbody>
</table>
## Task types

- **Types of tasks**

<table>
<thead>
<tr>
<th></th>
<th>HC</th>
<th>LC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>High criticality</td>
<td>Low criticality</td>
</tr>
<tr>
<td>Modes</td>
<td>HI and LO</td>
<td>no modes</td>
</tr>
<tr>
<td>WCET</td>
<td>Varies $C_i^h \geq C_i^l$</td>
<td>Static $C_i^h = C_i^l$</td>
</tr>
</tbody>
</table>

- **Modes for HC tasks**

<table>
<thead>
<tr>
<th></th>
<th>HI</th>
<th>LO</th>
</tr>
</thead>
<tbody>
<tr>
<td>WCET</td>
<td>$C_i^h$</td>
<td>$C_i^l$</td>
</tr>
</tbody>
</table>
Three scheduling functions

- Fixed priority scheduling
  
  \[ Sched : 2^\Pi \times \mathbb{R}_{\geq 0} \to \Pi \]

- Intermediate scheduling
  
  \[ Sched_I(\Pi, t) = \pi_i \mid \text{Ready}(\pi_i, t) \land \forall \pi_j \in \Pi \text{ Ready}(\pi_j, t) \Rightarrow \]
  
  \[ \begin{aligned}
  &\Omega(\pi_j, t) < \Omega(\pi_i, t) \\
  \lor
  \\
  &\Omega(\pi_j, t) = \Omega(\pi_i, t) \land Sched(\{\pi_i, \pi_j\}, t) = \pi_i
  \end{aligned} \]

- Critical scheduling
  
  \[ Sched_C(\Pi, t) = \pi_i \mid \text{Ready}(\pi_i, t) \land \forall \pi_j \in \Pi \text{ Ready}(\pi_j, t) \Rightarrow \]
  
  \[ \begin{aligned}
  &\chi_j < \chi_i \\
  \lor
  \\
  & (\chi_j = \chi_i) \land \Omega(\pi_j, t) < \Omega(\pi_i, t) \\
  \lor
  \\
  & (\chi_j = \chi_i) \land (\Omega(\pi_j, t) = \Omega(\pi_i, t)) \\
  \land
  \\
  & Sched(\{\pi_i, \pi_j\}, t) = \pi_i
  \end{aligned} \]
System scheduling mode behavior

- **Normal mode**: Workload (when all HC tasks run $C_h$) exceeds Threshold
- **Critical mode**: At least one HC task terminated
# Example

Simple mixed task set

<table>
<thead>
<tr>
<th>Task</th>
<th>T</th>
<th>$C^l$</th>
<th>$C^h$</th>
<th>$\chi$</th>
<th>$\rho$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\pi_1$</td>
<td>20</td>
<td>5</td>
<td>7</td>
<td>HC</td>
<td>2</td>
</tr>
<tr>
<td>$\pi_2$</td>
<td>20</td>
<td>5</td>
<td>6</td>
<td>HC</td>
<td>4</td>
</tr>
<tr>
<td>$\pi_3$</td>
<td>20</td>
<td>5</td>
<td>-</td>
<td>LC</td>
<td>1</td>
</tr>
<tr>
<td>$\pi_4$</td>
<td>20</td>
<td>4</td>
<td>-</td>
<td>LC</td>
<td>3</td>
</tr>
</tbody>
</table>
Example
Runtime example for the system in Table
Low criticality task behavior

- Active (regular)
- Active (shrunk period)
- Active (stretched period)

System scheduling-mode switched to Critical

All delays due to stretching absorbed

Period shrinking

Period stretching

System sched-mode switched to Critical

System sched-mode switched to Normal
High criticality task behavior

- Active (LO mode)
- WCET C_1 violation
- Period termination

- Active (HI mode)
Example of LC task periods shrinking

System mode:

- Critical Normal
- HI mode
- LO mode

HC task $\pi_1$:
(T=30, C=7)

LC task $\pi_2$:
(T=20, C=4)

Stretching

LC task $\pi_3$:
(T=14, C=3)

Shrinking

Shrinking with $\Delta=12$ over interval [5, 30]

Stretching

$\mu=6$

Shrinking

$\mu=4$
Algorithm 1: Elastic multimode scheduling

1. Init();
2. while True do
   3. if $\exists \pi_i \mid Status(\pi_i, t) = Done \land t \% T_i = 0$ then
      4. Refresh($\pi_i$);
   5. end
   6. if $\exists \pi_i \mid \chi_i = HC \land \Lambda(\pi_i, t) \geq C_i^t \land Status(\pi_i, t) \neq Done$ then
      7. $\Omega(\pi_i, t) = HI$;
      8. Use(Sched$_I$());
   9. end
10. if $Mode(t) = Normal \land DEM(lp_l(t), t) < lp_l(t).T - t \% lp_l(t).T$ then
    11. $T = lp_l(t)$;
    12. $S = t$;
    13. $Mode(t) = Critical$;
    14. $P = Stretching$;
    15. Use(Sched$_E$());
    16. foreach $\pi_j \mid \chi_j = LC$ do
       17. $T_j \leftarrow T_j + (lp_l(t).T - t \% lp_l(t).T)$;
       18. $\delta = \delta + (T.T - S)$;
    19. end
10. end
21. if $Mode(t) = Critical \land \exists \pi_i \mid T(\pi_i, S) \land t \% T_i = 0$ then
    22. $Mode(t) = Normal$;
    23. $P = Regular$;
    24. $\eta = t$;
    25. if $\exists \pi_j \mid \Omega(\pi_j, t) = HI$ then
       26. Use(Sched$_I$());
    27. end
    28. else
       29. Use(Sched$_D$());
    30. end
21. end
32. if $Mode(t) = Normal \land \delta > 0$ then
    33. if $DEM^S(lp_l(t), t) \leq lp_l(t).T - t$ then
       34. foreach $\pi_j \mid \chi_j = LC$ do
          35. $T_j = T_j - \mu_j$;
       36. end
       37. $P = Shrinking$;
       38. $\delta = 0$;
    39. end
41. end
Case study
Origin


- WCET \((C^l)\) given in original case.
- WCET \((C^h)\) calculated from data fetching times.
  - 20\(\mu s\) for data words
  - 40\(\mu s\) for a command
  - 40\(\mu s\) for a status
# Case study

## Tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>$\chi$</th>
<th>$T$</th>
<th>$C^l$</th>
<th>$C^h$</th>
<th>$\rho$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aircraft flight data ($\pi_1$)</td>
<td>HC</td>
<td>55</td>
<td>8</td>
<td>8.9</td>
<td>6</td>
</tr>
<tr>
<td>Steering ($\pi_2$)</td>
<td>HC</td>
<td>80</td>
<td>6</td>
<td>6.3</td>
<td>9</td>
</tr>
<tr>
<td>Target tracking ($\pi_3$)</td>
<td>HC</td>
<td>40</td>
<td>4</td>
<td>4.2</td>
<td>3</td>
</tr>
<tr>
<td>Target sweetening ($\pi_4$)</td>
<td>HC</td>
<td>40</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>AUTO/CCIP toggle ($\pi_5$)</td>
<td>HC</td>
<td>200</td>
<td>1</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>Weapon trajectory ($\pi_6$)</td>
<td>HC</td>
<td>100</td>
<td>7</td>
<td>7.5</td>
<td>10</td>
</tr>
<tr>
<td>Reinitiate trajectory ($\pi_7$)</td>
<td>LC</td>
<td>400</td>
<td>6.5</td>
<td>-</td>
<td>14</td>
</tr>
<tr>
<td>Weapon release ($\pi_8$)</td>
<td>HC</td>
<td>10</td>
<td>1</td>
<td>1.2</td>
<td>1</td>
</tr>
<tr>
<td>HUD display ($\pi_9$)</td>
<td>LC</td>
<td>52</td>
<td>6</td>
<td>-</td>
<td>7</td>
</tr>
<tr>
<td>MPD tactical display ($\pi_{10}$)</td>
<td>LC</td>
<td>52</td>
<td>8</td>
<td>-</td>
<td>8</td>
</tr>
<tr>
<td>Radar tracking ($\pi_{11}$)</td>
<td>HC</td>
<td>40</td>
<td>2</td>
<td>2.2</td>
<td>2</td>
</tr>
<tr>
<td>HOTAS bomb button ($\pi_{12}$)</td>
<td>LC</td>
<td>40</td>
<td>1</td>
<td>-</td>
<td>5</td>
</tr>
<tr>
<td>Threat response display ($\pi_{13}$)</td>
<td>LC</td>
<td>100</td>
<td>3</td>
<td>-</td>
<td>11</td>
</tr>
<tr>
<td>Poll RWR ($\pi_{14}$)</td>
<td>LC</td>
<td>200</td>
<td>2</td>
<td>-</td>
<td>13</td>
</tr>
<tr>
<td>Perodic BIT ($\pi_{15}$)</td>
<td>LC</td>
<td>1000</td>
<td>5</td>
<td>-</td>
<td>15</td>
</tr>
</tbody>
</table>
Case Study
Uppaal model
Case Study

Results

- Not schedulable with classical fixed priority scheduling
  - tasks $\pi_{10}$ and $\pi_{11}$ miss their deadlines
- Not schedulable with task-level mode scheduling
  - task $\pi_{10}$ misses its deadline (response time 106)
- System-level scheduling vs. our algorithm
Case study
Experimental results

- Our multimode algorithm
- System-level mode algorithm

Graph showing LC jobs discarded over simulation time (1000 s).
Discard rate of the LC task jobs achieved by our algorithm is 1.0% to 4.58%.

Discard rate achieved by the state of the art system-level bi-mode scheduling [13], [33] is 2.1% to 11.5%.


Conclusion

- Flexible multi-mode scheduling for mixed-criticality systems
  - accurate and non-aggressive system mode switches
- Stretching of periods
- Much less dropping of LC tasks
- Too computation heavy at the moment
Conclusion

And future work

Conclusion

▶ Flexible multi-mode scheduling for mixed-criticality systems
  ▶ accurate and non-aggressive system mode switches
▶ Stretching of periods
▶ Much less dropping of LC tasks
▶ Too computation heavy at the moment

Future Work

▶ Real implementation
▶ Optimization of algorithm overhead
Questions?

System scheduling-mode switched to Critical

Shrinking with $\Delta=12$ over interval $[5,30]$

System mode: Critical Normal

HI mode

LC task $\pi_2$: (T=20, C=4)

Shrinking

Stretching Regular

System mode:

LO mode

Critical

Normal

Stretching

Shrinking

Active (regular)

Active (stretched period)

Active (shrunk period)

All delays due to stretching absorbed

System sched-mode switched to Critical

System sched-mode switched to Normal

Active (regular)

Active (stretched period)

Active (shrunk period)

Period shrinking

Period stretching

System scheduling-mode switched to Critical

Our multimode algorithm

System-level mode algorithm

$\mu=6$

$\mu=4$

$\Delta=12$

$[5,30]$

$T=30, C=7$

$T=20, C=4$

$T=14, C=3$