Overview

- Controller Area Network (CAN)
- Scheduling model (recap)
- Existing analysis for CAN (recap)
- Examples refuting existing analysis
- Analysis based on discrete scheduling
- What’s new for analysis for $D \leq T$?
- Conclusion

Controller Area Network (CAN)

- a serial, broadcast bus;
- short real-time control messages (0 – 8 bytes);
- speed: up to 1 Mbit/sec
- usage:
  - originally: automotive industry
  - today: numerous industrial applications
- documentation:
  - CAN specification, Bosch, September 1991
  - ISO 11898 road vehicles, November 1993

Analysis for CAN

- CAN bus
  - messages are non-preemptive;
  - fixed priorities;
- Analysis
  - Fixed-Priority Non-preemptive Scheduling (FPNS)
  - academic literature: >300 refs
  - industrial practice: tool vendors
  - but … analysis for FPNS is flawed (optimistic)

Scheduling model for FPNS (recap)

- Based on model for FPNS
  - Single processor;
  - Set of $n$ independent periodic tasks $\tau_1, \tau_2, \ldots, \tau_n$;
  - Continuous scheduling;
  - Characteristics of task $\tau_i$;
    - (release) period $T_p$
    - computation time $C_i$
    - (relative) deadline $D_i$ (where $D_i \leq T_i$);
    - phasing $\phi_i$;
  - Basic assumptions:
    - similar to [Liu and Layland 73], but …
**Model for FPNS (recap)**

- Refinement for FPNS:
  - Each job of task \( \tau_i \) is non-preemptive;
- Notational convenience:
  - tasks in order of decreasing priority;
  - i.e. \( \tau_1 \) highest and \( \tau_n \) lowest priority.

**Existing analysis for FPNS/CAN (recap)**

- Blocking due to non-preemptive nature
  \[ B_i = \max_{j \neq i} C_j \]
- worst-case message response time
  \[ \bar{R}_i^N = w_i + C_i \]
  where \( w_i \) is the smallest positive value satisfying
  \[ x = B_i + \sum_{j \neq i} \frac{x + \tau_{ij}}{T_j} C_j \]
  and \( \tau_{ij} = 1 \mu s \) for a 1Mbit/sec CAN-bus.

**FPNS example refuting existing analysis**

<table>
<thead>
<tr>
<th>Task</th>
<th>Period ( T_i )</th>
<th>Computation time ( C_i )</th>
<th>Deadline ( D_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \tau_1 )</td>
<td>10</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>( \tau_2 )</td>
<td>15</td>
<td>3</td>
<td>15</td>
</tr>
<tr>
<td>( \tau_3 )</td>
<td>15</td>
<td>4</td>
<td>13</td>
</tr>
</tbody>
</table>

Existing analysis:
\[ \bar{R}_3^N = 12 \]

Conclusion:
Existing worst-case analysis for FPDS/FPNS is optimistic!

**CAN example refuting existing analysis**

<table>
<thead>
<tr>
<th>Message</th>
<th>Period ( T )</th>
<th>Computation time ( C )</th>
<th>( \bar{R}_i^N ) (16)</th>
<th>( \bar{R}_i^N ) (6)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mu_1 )</td>
<td>214</td>
<td>85</td>
<td>160</td>
<td>159</td>
</tr>
<tr>
<td>( \mu_2 )</td>
<td>289</td>
<td>65</td>
<td>225</td>
<td>224</td>
</tr>
<tr>
<td>( \mu_3 )</td>
<td>290</td>
<td>75</td>
<td>280</td>
<td>299</td>
</tr>
<tr>
<td>( \mu_4 )</td>
<td>3000</td>
<td>55</td>
<td>590</td>
<td>590</td>
</tr>
</tbody>
</table>

**Analysis based on discrete scheduling [6]**

- Blocking due to non-preemptive nature
  \[ B_i = \max_{j} (C_j - 1) \]
- worst-case message response time
  \[ \bar{R}_i^N = \max_{j} (w_{ij} C_j + C_i - qT_i) \]
  where \( w_{ij} \) is the smallest positive value satisfying
  \[ x = B_i + qC_i + \sum_{j} \left( 1 + \frac{x}{T_j} \right) C_j \]
- \( Q = \left\lfloor \frac{x}{T_i} \right\rfloor \) and \( l_i \) is the smallest positive value satisfying
  \[ l_i = B_i + \sum_{j} \frac{1}{T_j} C_j \]

**What’s new for analysis for \( D \leq T \)?**

- Analysis for deadlines at most equal to periods: “thy shall look beyond the first job”
  - for FPPS and tasks with varying priorities [Gonzalez Harbour et al. 94];
  - for preemption threshold scheduling [Regehr 02];
  - for FPDS [Bril 06];
  - for FPNS and CAN (this presentation);
  - for EDF [Spuri 96].
- Perhaps it’s time to consider analysis for FPPS the exception rather than the rule…
- and teach students accordingly.
Conclusion

- Problem resolved by discrete scheduling [6]
- Current work for CAN:
  - Can arbitrary message fail?
  - How about jitter and errors?
  - Guidelines for tool vendors and industry?
  - How to increase "similarity"?
  - How to handle "end-to-end" issues?
- Status:
  - Joint York & TU/e CS-report under construction.
- Future work:
  - FPNS for continuous scheduling.

Additional references