Exercise Sheet 7
(18 Punkte)

Exercise Due at Wednesday, June 26, 2019, 12:00 Uhr


Note: It is allowed to work in a group of up to three persons, if these persons are from the same practice group. Please do not forget to write your name and your Matrikelnummer on the solutions. The solutions can either be placed in the marked mailbox in front of the secretary's office of LS 12 (OH16/E22) or sent by mail (PDF format) to georg.von-der-brueggen [✉️] tu-dortmund.de

Definitions: Assume the tasks $\tau_i \in \tau$ are ordered according to Rate Monotonic and that all tasks have different periods, i.e. $T_i \neq T_j \forall \tau_i, \tau_j \in \tau$ with $i \neq j$

- $hp(\tau_k)$ consists of the $\tau_i \in \tau$ with $T_i < T_k$, i.e., all tasks with higher priority then $\tau_k$
- $lp(\tau_k)$ consists of the $\tau_i \in \tau$ with $T_i > T_k$, i.e., all tasks with lower priority then $\tau_k$
- Blocking time: $B_k = \max_{\tau_i \in lp(\tau_k)} \{C_i\}$
- $t_k$ is the deadline of $T_k$
- $\{t_1, \ldots, t_k\}$ are the last releases of the tasks $\tau_i \in hp(\tau_k)$ before $t_k$ and $t_k$ itself

7.1 $k^2 U$ Framework (4 Punkte)

1. Use the $k^2 U$ Framework to prove the Liu and Layland Bound for implicit deadline task sets under Rate Monotonic Scheduling
   \[
   \sum_{i=1}^{k} U_i \leq k(2^\frac{1}{k} - 1)
   \]

2. A sufficient schedulability test for non-preemptive rate monotonic scheduling is to show, that $\exists t_j \in \{t_1, \ldots, t_k\}$ such that
   \[
   C_k + B_k + \sum_{i=1}^{k-1} \left\lceil \frac{t_j}{T_i} \right\rceil C_i \leq t_j
   \]
   if the schedulability of all higher priority tasks was already guaranteed.
   Show that a task $\tau_k$ in a non-preemptive sporadic task system with implicit deadlines can be feasibly scheduled by the Rate Monotonic scheduling algorithm, if the schedulability for all higher priority tasks has already been ensured and the following condition holds:
   \[
   \left( \frac{C_k + B_k}{T_k} + 1 \right) \prod_{\tau_j \in hp(\tau_k)} (U_j + 1) \leq 2
   \]
7.2 Multi-Frame Task Systems (4 Punkte)

Given \( n \) independent, preemptable, and periodic tasks with implicit deadlines, there are two types of executions for each task \( \tau_i \). It is known that \( C_{i,2} \leq C_{i,1} \leq 2C_{i,2} \), in which \( C_{i,1} \) is the WCET of the first version, and \( C_{i,2} \) is the WCET of the second version.

Suppose that the tasks are ordered in a rate-monotonic order, in which \( T_i \leq T_{i+1} \) for \( i = 1, 2, \ldots, n - 1 \) and \( 2T_i \geq T_n \). For any two consecutive jobs of task \( \tau_i \), at most one of them is executed by using the first-version (i.e., \( C_{i,1} \)).

The task set is schedulable under rate monotonic scheduling if

\[
\prod_{i=1}^{n} \left( \frac{C_{i,2}}{T_i} + 1 \right) \leq 1.5.
\]

Prove the above statement, and also generalize the above bound for another condition \( C_{i,2} \leq C_{i,1} \leq \alpha C_{i,2} \) where \( \alpha \geq 1 \).

7.3 Communication CAN Bus (2 Punkte)

Consider the following case with four sporadic messaging tasks on a CAN. Suppose that the worst-case transmission time of a message is at most 200 \( \mu s \). Each sporadic messaging task sends only one message per job.

<table>
<thead>
<tr>
<th>symbol</th>
<th>identifier</th>
<th>minimum inter-arrival time and relative deadline (( \mu s ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>000000000001</td>
<td>500</td>
</tr>
<tr>
<td>B</td>
<td>000000001011</td>
<td>800</td>
</tr>
<tr>
<td>C</td>
<td>000000011111</td>
<td>2000</td>
</tr>
<tr>
<td>D</td>
<td>000000000110</td>
<td>1400</td>
</tr>
</tbody>
</table>

Can a message of a task be sent before its relative deadline? If not, evaluate whether there exists a feasible schedule to send the messages for these tasks (without changing the timing parameters) on the CAN.
7.4 RTOS and RTPL (3 Punkte)

1. What are the advantages to use Real-Time Programming Languages and Real-Time Operating Systems to design real-time systems, respectively?

2. Explain the concepts behind Delay and Delay until in Ada. Check the source code of FreeRTOS. What are the corresponding functions in FreeRTOS?

3. Explain why most commercial Real-Time Operating Systems use only a fixed-length table to store the (pointers of) task control blocks (TCB). How does the scheduler in FreeRTOS work?

Hint: 1+1+1

7.5 RTC (4 Punkte)

1. Mr. Smart wants to define the feasibility of his design as follows: “the system is feasible if $\alpha^a(\Delta) \leq \beta^l(\Delta)$ for any $\Delta \geq 0$”, where $\alpha$ and $\beta$ are arrival curve and service curve, respectively. Is this a reasonable definition?

2. How to derive the arrival curves from a set of given traces of events? How to derive the service curves from a set of traces of services?

3. What are the limitations of using Greedy Processing Components (GPC) in Real-Time Calculus (RTC)? How to perform schedulability analysis for fixed-priority scheduling based on GPCs? For fixed-priority scheduling, what are the similarity and difference of worst-case response time analysis in RTC to and from the time-demand schedulability tests for sporadic real-time tasks with arbitrary deadlines?

Hint: 1+1+2

7.6 Real-Time Calculus (1 Punkt)

Is the arrival curve in the left correct? Is it tight? If yes, why? If not, how to make it tighter?