11.1 Modular Performance Analysis with Real-Time Calculus – Modeling and Analysis of Systems

For this exercise you need to use the MATLAB MPA/RTC Toolbox as in Exercise 10. Reminder: You need to run the script rtc_load.m before you can use the RTC Toolbox.

11.1.a) Event Streams

Consider a task $T$ where jobs arrive with a period 10, a maximum jitter 15, a minimum inter-arrival distance 2, and have an execution time of 1. Consider a specific trace of these jobs which arrive at times 0, 10, 18, 21, 33, 40, 55, 65, 75, 90, 100, 120. Plot the input arrival function (or the input event stream) $R(t)$ using the commands rtccurve and rtcplot.

Now plot the arrival curve $\alpha$ of this task using the commands rtcpjd and rtcplot. Extract and plot only the upper-arrival curve $\alpha^u$.

Test if the arrival times of the jobs used to construct $R(t)$ conform to the specification of the upper-arrival curve $\alpha^u$. Hint: Use the rtcmindeconv operator which is defined as follows

\[
(f \circ g)(\Delta) = \sup_{\lambda \geq 0} \{f(\Delta + \lambda) - g(\lambda)\}. \tag{1}
\]

**Solution:**

The arrival function is computed and plotted as follows.

\[
R = \text{rtccurve}([[0 1 0];[10 2 0];[18 3 0];[21 4 0];[33 5 0];[40 6 0];...[55 7 0];[65 8 0];[75 9 0];[90 10 0];[100 11 0];[120 12 0]]);
\]

\[
\text{rtcplot}(R, 120);
\]

The arrival curve is computed and plotted as follows

\[
\alpha = \text{rtcpjd}(10, 15, 2);
\]

\[
\text{rtcplot}(\alpha)
\]

The upper arrival curve is obtained as $\alpha^u$.

The conformance to the upper-arrival curve can be checked by plotting $(R \circ R)$ and $\alpha^u$. This is done as follows.

\[
S = \text{rtcmindeconv}(R, R);
\]

\[
\text{rtcplot}(S, 'r', \alpha^u, 'b')
\]
The red line should not exceed the blue line at any point.

11.1.b) Resource Streams

Consider a bus with TDMA arbitration with slot-size 1 and period 8. Consider a specific resource availability trace where the TDMA slot is available in the interval $[4, 5]$. Plot the service function (or resource stream) $C(t)$ using the commands rtccurve and rtcplot.

Now plot the service curve $\beta$ of this task using the commands rtctdma and rtcplot. Set the bandwidth parameter to 1. Extract and plot only the lower-arrival curve $\beta_l$.

Test if the plotted $C(t)$ conforms to the specification of the lower-service curve $\beta_l$. Hint: Use the rtcmaxdeconv operator which is defined as follows

$$(f \ominus g)(\Delta) = \inf_{\lambda \geq 0} \{f(\Delta + \lambda) - g(\lambda)\}. \quad (2)$$

Solution:

The service function is computed and plotted as follows.

```matlab
C = rtccurve([[0,0,0]], [[0,0,1];[1,1,0]], 4, 0, 8, 1);
rtcplot(C, 30);
```

The service curve is computed and plotted as follows.

```matlab
beta = rtctdma(1, 8, 1);
rtcplot(beta, 30);
```

The lower service curve is obtained as beta(2).

The conformance to the lower service curve can be checked by plotting $(C \ominus C)$ and $\beta_l$. This is done as follows.

```matlab
D = rtcmaxdeconv(C, C);
rtcplot(D, 'r', beta(2), 'b')
```

The blue line should not exceed the red line at any point.

11.1.c) Output Streams

For the event stream $R(t)$ and the resource stream $C(t)$ compute the output event stream $R'(t)$ using the following definition (derived in lecture slide 11-27)

$$R'(t) = \inf_{0 \leq u \leq t} \{ R(u) + C(t) - C(u) \}. \quad (3)$$

Plot $R'(t)$ using the commands rtccurve and rtcplot.

Compute the maximum delay and maximum backlog using $R(t)$ and $R'(t)$. Hint: Use the commands rtch and rtcv.

Confirm that the following condition (lecture slide 11-34) holds using the command rtcminconv.

$$R'(t) \geq (R \otimes \beta')(t). \quad (4)$$

Solution:

The output arrival function is computed plotted as follows.
R_prime = rttccurve([[0,0,0];[4,0,1];[5,1,0];[12,1,1];[13,2,0];...
[20,2,1];[21,3,0];[28,3,1];[29,4,0];[36,4,1];[37,5,0];...
[44,5,1];[45,6,0];[60,6,1];[61,7,0];[68,7,1];[69,8,0];...
[76,8,1];[77,9,0];[92,9,1];[93,10,0];[100,10,1];[101,11,0];...
[124,11,1];[125,12,0]]);
rtcplot(R, 'r', C, 'b', R_prime, 'g', 120);

The maximum backlog and delay are computed as follows
max_del = rtch(R, R_prime); % output value = 8
max_buf = rtcv(R, R_prime); % output value = 1

The said condition is validated by checking the following plot.
rtcplot(R_prime, 'r', rtcminconv(R, beta(2), 'b'));

The red line should always exceed the blue line.

11.1.d) Output Curves
Compute the worst-case bounds on the delay and backlog using $\alpha^u$ and $\beta^i$. Hint: Use the commands rtch and rtcv. You may visualize this with the commands rtcplot and rtcplotv.
Plot the output arrival curve $\alpha'$ and the output service curve $\beta'$ using the command rtcgpc. This command also gives you the worst-case bounds on the delay and backlog.

**Solution:**
The bounds are computed as follows.
del = rtch(alpha(1), beta(2)); % output = 19
buf = rtcv(alpha(1), beta(2)); % output = 3

The output curves are plotted as follows.
[alpha_prime beta_prime del buf] = rtcgpc(alpha, beta, 1);

11.1.e) Modular Analysis of Multiple Components
Let the output of the TDMA be executed on another resource, say a processor. Each job of task $T$ has an execution time of 3 on this processor. This processor also executes a different task $T_1$ with a higher priority under fixed priority scheduling. $T_1$ has a period 5, jitter 1, and an execution time 2.

This system can be visualized by the block diagram as shown in Figure 1. The goal is to plot the output arrival curve $\alpha''$ as indicated in the figure. To this end, first compute the service curve of the processor $\beta_{CPU}$ using the command rtcfs, with a bandwidth parameter 1. This models a processor with full service. Then model the arrival curve of task $T_1$ ($\alpha_1$ in the figure). Then, compute the available service to $GPC'$ ($\beta'_{1}$ in the figure) command rtcgpc for $GPC_1$. Finally, compute $\alpha''$ with the command rtcgpc for $GPC''$.

Compute the buffer size required at the input of $GPC'$. Compute the worst-case end-to-end delay for task $T$.

**Solution:**

3
Figure 1: Block diagram for task 9.1e)

\[
\beta_{\text{cpu}}
\]

\[
\beta_{\text{bus}}
\]

\[
\text{Fixed Priority}
\]

\[
\text{TDMA}
\]

\[
\alpha_1 \rightarrow \text{GPC}_1
\]

\[
\alpha \rightarrow \text{GPC}
\]

\[
\alpha' \rightarrow \text{GPC}'
\]

\[
\alpha''
\]

\[
\beta_1
\]

\[
\text{end_to_end_delay} = \text{del} + \text{del_prime}; \ % \text{output value} = 19 + 7 = 26
\]

11.1.f) Design Space Exploration

Increase the execution time of task \( T_1 \) from 2 to 4. What is the effect on the worst-case end-to-end delay of task \( T \)?

What is the largest value of the execution time of task \( T_1 \) such that the end-to-end deadline of 50 is satisfied for \( T \)?

**Solution:**

When the execution time of \( T_1 \) is increased to 4, the delay in \( GPC' \) increases to \( \text{Inf} \). This is because no resource is remaining on the processor after executing \( T_1 \).

The largest value of the execution time of \( T_1 \) that meets the end-to-end deadline of \( T \) is about 3.46.
11.2 Homework

The following tasks are intended for further practice with the toolbox.

11.2.a) Pay-Bursts-Only-Once

Consider the system depicted in Fig. 2. A bursty event stream is processed by a sequence of three tasks with increasing execution time. The tasks are running on independent CPUs which have full services with bandwidth \( \lambda = 1 \).

![Diagram of system](image)

**Figure 2: Specification of system 2**

**Input stream**
- Periodic with burst \((P = 10ms, J = 50ms, D = 1ms)\)

**Execution times**
- T1: 1ms, T2: 4ms, T3: 8ms

a) Compute the worst-case end-to-end delay I1-O1 as sum of the worst-case delays at T1, T2 and T3 (call the `del` function 3 times)

b) Compute the worst-case end-to-end delay I1-O1 using the delay analysis for consecutive GPCs (call the `del` function only once)

c) Try to explain the different results obtained for a) and b)

**Solution:**

a) Sum of the individual delays: 60ms

b) Optimized delay analysis: 51ms

c) Compositional analysis assumes that the worst case scenario for all components can coincide. Therefore, a compositional approach where we sum-up the individual delays from each task is more pessimistic than a holistic (global) analysis of the system.

```matlab
% Setup of system parameters

% Event stream
p = 10;
j = 50;
```
\[d = 1;\]

% Worst-case execution demands

\[vced1 = 1;\]
\[vced2 = 4;\]
\[vced3 = 8;\]

% Processor speeds

\[cpu1 = 1;\]
\[cpu2 = 1;\]
\[cpu3 = 1;\]

% Construct input curves
\[
\text{\texttt{\texttt{a1i = rtcjd(p,j,d);}}} \]
\[
\text{\texttt{\texttt{b1i = rtcfs(cpu1);}}} \]
\[
\text{\texttt{\texttt{b2i = rtcfs(cpu2);}}} \]
\[
\text{\texttt{\texttt{b3i = rtcfs(cpu3);}}} \]

% Analysis.

% Compute the arrival and service curves through a chain of Greedy Processing Components.
\[
\text{\texttt{\texttt{\texttt{[a1o b1o] = rtcgpc(a1i, b1i, vced1);}}} \]
\[
\text{\texttt{\texttt{[a2o b2o] = rtcgpc(a1o, b2i, vced2);}}} \]
\[
\text{\texttt{\texttt{[a3o b3o] = rtcgpc(a2o, b3i, vced3);}}} \]
\]
% Compute the total delay as the sum from the individual delays.
\[
\text{\texttt{\texttt{\texttt{\texttt{delay_add = rtcdel(a1i,b1i,vced1) + rtcdel(a1o,b2i,vced2) + rtcdel(a2o,b3i,vced3);}}} \]
\]
% Compute the total delay through service curve convolution.
\[
\text{\texttt{\texttt{\texttt{\texttt{delay_opt = rtcdel(a1i,b1i,vced1,b2i,vced2,b3i,vced3);}}} \]
\]
% Display the results.
\[
\text{\texttt{\texttt{\texttt{\texttt{disp("Total delay (sum of the individual delays) : "} \num2str(delay_add)");}}} \]
\[
\text{\texttt{\texttt{\texttt{\texttt{disp("Total delay (optimized delay analysis) : "} \num2str(delay_opt)");}}} \]
\]
11.2.b) Hierarchical Scheduling

Given is a simple system for car engine control with hierarchical scheduling on the communication bus. The system has two engine control units (ECUs) communicating through a bus with Time Division Multiple Access (TDMA) arbitration. The communication on the bus is achieved with two separate slots in TDMA policy. The slots can be used by different types of messages. The messages are assigned priorities when more than one message is available to use a given slot. There are three sensors generating incoming data and there are seven processing tasks mapped to the two ECUs. The specification of the system is given in Tables 1, 2a, 2b, and 2c. You can consider that each ECU has a full processor available at the speed of 5000 cycles per second.

The system mapping of tasks and messages is given in Figure 3. The scheduling of the tasks on the two ECUs is organized with fixed priority.

a) Model the system in MPA and find the total end-to-end delays for all streams. You can consider that there is no interference from other tasks (OS, other applications, etc.) and context switches take negligible time.

NOTE: You can work in milliseconds (ms) e.g. the ECU speeds become 5 cycles per ms.

b) At the current moment the communication bus is busy 100% of the time but most of its bandwidth is wasted (the two slots take all of the cycle length). Find alternative parameters for the cycle and slot lengths of the TDMA which still satisfy the deadline requirements of the streams.

Table 1: Specifications of the three data streams generated by the sensors.

<table>
<thead>
<tr>
<th>Streams</th>
<th>(p,j,d) [ms]</th>
<th>Deadline [ms]</th>
<th>Task Chain</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>(10, 30, 5)</td>
<td>6</td>
<td>T1.1 -&gt; C1.1 -&gt; T2.1</td>
</tr>
<tr>
<td>s2</td>
<td>(20, 40, 5)</td>
<td>16</td>
<td>T1.2 -&gt; C1.2 -&gt; T2.2 -&gt; C2.1 -&gt; T1.3</td>
</tr>
<tr>
<td>s3</td>
<td>(50, 0, 0)</td>
<td>17</td>
<td>T2.3 -&gt; C2.2 -&gt; T1.4</td>
</tr>
</tbody>
</table>

Solution:

a) 2.9 ms, 8.5 ms, 8.8 ms
Table 2: Parameters of the system.

(a) Specifications of the worst case execution demands of the tasks.

<table>
<thead>
<tr>
<th>Task</th>
<th>WCED [cycles]</th>
</tr>
</thead>
<tbody>
<tr>
<td>T11</td>
<td>4</td>
</tr>
<tr>
<td>T12</td>
<td>4</td>
</tr>
<tr>
<td>T13</td>
<td>5</td>
</tr>
<tr>
<td>T14</td>
<td>5</td>
</tr>
<tr>
<td>T21</td>
<td>5</td>
</tr>
<tr>
<td>T22</td>
<td>5</td>
</tr>
<tr>
<td>T23</td>
<td>5</td>
</tr>
</tbody>
</table>

(b) Specifications of the length of communication messages.

<table>
<thead>
<tr>
<th>Message</th>
<th>length [bits]</th>
</tr>
</thead>
<tbody>
<tr>
<td>C11</td>
<td>10</td>
</tr>
<tr>
<td>C12</td>
<td>10</td>
</tr>
<tr>
<td>C21</td>
<td>10</td>
</tr>
<tr>
<td>C22</td>
<td>10</td>
</tr>
</tbody>
</table>

(c) Specification of the communication bus.

<table>
<thead>
<tr>
<th>TDMA</th>
<th>—</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bandwidth</td>
<td>100 bits/ms</td>
</tr>
<tr>
<td>Cycle</td>
<td>2 ms</td>
</tr>
<tr>
<td>Slot CC1</td>
<td>1 ms</td>
</tr>
<tr>
<td>Slot CC2</td>
<td>1 ms</td>
</tr>
</tbody>
</table>

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% example for a system with Hierarchical Scheduling
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % 2 x ECUs with fixed priority
% % 1 x TDMA communication bus with 2 communication channels
% % Each comms channel with fixed priority internally
% % 7 tasks
% % 3 event streams

% streams
s1 = rtcpjd(10, 30, 5);
s2 = rtcpjd(20, 40, 5);
s3 = rtcpjd(50, 0, 0);

% ECUs
ecu1 = rtcfs(5);
ecu2 = rtcfs(5);

% bus
c1 = rtctdma(1, 2, 100);
c2 = rtctdma(1, 2, 100);

% WCED for tasks
t11 = 4;
t12 = 4;
t13 = 5;
t14 = 5;
t21 = 5;
t22 = 5;
t23 = 5;

% message lengths
c11 = 10;
c12 = 10;
c21 = 10;
c22 = 10;

% task chain for S1
[sT11 bT11] = rtcgpc(s1, ecu1, t11);
[sC11 bC11] = rtcgpc(sT11, cc1, c11);
[sT21 bT21] = rtcgpc(sC11, ecu2, t21);

% task chain for S2
[sT12 bT12] = rtcgpc(s2, bT11, t12);
[sC12 bC12] = rtcgpc(sT12, bC11, c12);
[sT22 bT22] = rtcgpc(sC12, bT21, t22);
[sC21 bC21] = rtcgpc(sT22, cc2, c21);
[sT13 bT13] = rtcgpc(sC21, bT12, t13);

% task chain for S3
[sT23 bT23] = rtcgpc(s3, bT22, t23);
[sC22 bC22] = rtcgpc(sT23, bC21, c22);
[sT14 bT14] = rtcgpc(sC22, bT13, t14);

rtcdel(s1, ecu1, t11, cc1, c11, ecu2, t21)
rtcdel(s2, bT11, t12, bC11, c12, bT21, t22, cc2, c21, bT12, t13)
rtcdel(s3, bT22, t23, bC21, c22, bT13, t14)

b) e.g. Cycle length 5 ms, slot CC1 1 ms, slot CC2 1 ms, and other possibilities.