Solutions to the exam in Real-Time Systems 190829

These solutions are available on WWW:
http://www.control.lth.se/course/FRTN01/

1.

a.

$$\Phi(h) = e^{Ah} = \sum_{k\geq 0} \frac{1}{k!} A^k h^k = I + Ah = \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix}$$

$$\Gamma(h) = \int_0^h e^{At} B \, dt = \int_0^h \begin{bmatrix} t \\ 1 \end{bmatrix} dt = \begin{bmatrix} t^2/2 \\ t \end{bmatrix} \Bigg|_{t=0}^{h} = \begin{bmatrix} h^2/2 \\ h \end{bmatrix}$$

$$x(k+1) = \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} h^2/2 \\ h \end{bmatrix} u(k).$$

The system then becomes

$$x(k+1) = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 2 \\ 2 \end{bmatrix} u(k).$$

b. The continuous-time poles are in 0 and are mapped by the ZOH to two poles in 1. Those poles describe double integrator dynamics in both the time domains, this is expected since the ZOH doesen't alterate such dynamic $z_i = e^{h \cdot s_i}$.

2.

a. In the original structure, $C(z)$ is responsible for handling both set-point changes and disturbances. This means that there is a trade off in the performance. In the new structure, the problems have been separated. $C(z)$ now handles disturbances and $G_m(z)$ and $G_{ff}(z)$ handles set-point changes. This gives one more degree of freedom and therefore often better performance.

b.

$$Y(z) = \frac{P(G_{ff}(z) + C(z)G_m(z))}{1 + P(z)C(z)} R(z).$$

3.

a. The cont. transfer function is given by

$$G(s) = C(sI - A)^{-1} B = \frac{1}{s^2 + 6s + 9} = \frac{1}{(s+3)^2}.$$

So the poles are located in $s = -3$ and the system is stable.

b. With forward difference the approximation is given by $H(z) = G(s')$ with $s' = z - 1$.

$$H(z) = \frac{1}{(s'+3)^2} = \frac{1}{(z-1+3)^2} = \frac{1}{(z+2)^2}.$$

So the poles are located in $z = -2$, which is outside the unit circle, and the discrete-time system is unstable. (Sample time $h = 1$)

c. With Tustin approximation we have that $s' = \frac{2}{h} \cdot \frac{z-1}{z+1}$, with $h = 1$ we get:

$$\begin{aligned} H(z) = G(s') &= \frac{1}{\left(2\frac{z-1}{z+1} + 3\right)^2} = \frac{1}{\left(2\frac{z-1}{z+1}\right)^2 + 6 \cdot 2\frac{z-1}{z+1} + 9} \\ &= \frac{(z+1)^2}{4(z-1)^2 + 12(z-1)(z+1) + 9(z+1)^2} \\ &= \frac{(z+1)^2}{4z^2 - 8z + 4 + 12z^2 - 12 + 9z^2 + 18z + 9} \\ &= \frac{(z+1)^2}{25z^2 + 10z + 1} = \frac{1}{25} \cdot \frac{(z+1)^2}{z^2 + \frac{10}{25}z + \frac{1}{25}} \\ &= \frac{0.4(z+1)^2}{z^2 + 0.4z + 0.04} = \frac{0.4(z+1)^2}{(z+0.2)^2}. \end{aligned}$$

So there is a zero at $z = -1$ and two poles at $z = -0.2$, which is within the unit circle and hence the system is stable.

4.

a. A continuous-time double pole in $s = -1$ corresponds to a discrete-time double pole in $e^{-h} = e^{-0.1}$, i.e., the desired closed loop characteristic polynomial should be

$$(z - e^{-0.1})^2 = z^2 - 2e^{-0.1}z + e^{-0.2} = z^2 + p_1 z + p_2.$$

With the linear feedback

$$u(k) = -l_1 x_1(k) - l_2 x_2(k)$$

the closed-loop system becomes

$$x(k+1) = \begin{pmatrix} 1 - l_1 h^2/2 & h - l_2 h^2/2 \\ -l_1 h & 1 - l_2 h \end{pmatrix} x(k).$$

The characteristic polynomial of the closed-loop system is

$$z^2 + \left(\frac{l_1 h^2}{2} + l_2 h - 2\right)z + \left(\frac{l_1 h^2}{2} - l_2 h + 1\right).$$

Comparing this with the desired characteristic polynomial leads to the following linear equations for l_1 and l_2

$$\begin{aligned} \frac{h^2 l_1}{2} + h l_2 - 2 &= p_1 \\ \frac{h^2 l_1}{2} - h l_2 + 1 &= p_2 \end{aligned}$$

with the solution

$$\begin{aligned} l_1 &= \frac{1}{h^2}(1 + p_1 + p_2) = 0.9056 \\ l_2 &= \frac{1}{2h}(3 + p_1 - p_2) = 1.8580 \end{aligned}.$$

b. The characteristic polynomial of the observer is given by

$$\begin{aligned}\det(zI - \Phi + KC) &= \det\begin{pmatrix} z - 1 + k_1 & -0.1 \\ k_2 & z - 1 \end{pmatrix} \\ &= z^2 + (k_1 - 2)z + 1 - k_1 + 0.1k_2\end{aligned}.$$

The desired characteristic polynomial is

$$(z - e^{-0.2})^2 = z^2 - 2e^{-0.2}z + e^{-0.4}$$

Equating the coefficients we get

$$\begin{cases} k_1 - 2 = -2e^{-0.2} \\ 1 - k_1 + 0.1k_2 = e^{-0.4} \end{cases} \Rightarrow \quad K = \begin{pmatrix} 0.3625 & 0.3286 \end{pmatrix}^T.$$

c. The model and feedforward generator design is based is performed as follows. In order to make sure that the states of the model are compatible with the states of the process the following approach can be used. To begin with the model can be chosen identical to the process, i.e.,

$$\begin{aligned} x_m(k+1) &= \Phi x_m(k) + \Gamma u_{ff}(k) \\ y_m(k) &= C x_m(k) \end{aligned}. \tag{1}$$

The dynamics of the model can then be modified by the linear control law

$$u_{ff}(k) = -L_m x_m(k) + l_r u_c(k). \tag{2}$$

The model dynamics is then given by

$$\begin{aligned} x_m(k+1) &= (\Phi - \Gamma L_m) x_m(k) + \Gamma l_r u_c(k) \\ y_m(k) &= C x_m(k) \end{aligned}. \tag{3}$$

Here, L_m is chosen to give the model the desired eigenvalues and l_r is chosen to give the model a static gain of 1. The feedforward control signal $u_{ff}(k)$ is generated in such a way that it will give the desired behavior when used as an input to the process.

The desired characteristic polynomial of the model is given by

$$(z - e^{-2h})^2 = (z - e^{-0.2})^2 = z^2 - 2e^{-0.2}z + e^{-0.4}$$

From the first sub-problem it follows that the coefficients of L_m should be chosen as

$$\begin{aligned} l_1 &= \frac{1}{h^2}(1 + p_1 + p_2) = 3.2859 \\ l_2 &= \frac{1}{2h}(3 + p_1 - p_2) = 3.4611 \end{aligned}$$

Finally, l_r is chosen to get the static gain 1. This is obtained by setting

$$l_r = \frac{1}{C(I - \Phi + \Gamma L_m)^{-1}\Gamma} = 3.2$$

d. Both the observer and the process model should be updated in UpdateState. Also in UpdateState pre-calculations can be done both for u and for u_{ff}. This leads to the following pseudo-code:

CalculateOutput:
Sample y and obtain u_c
$u_{ff} = u_{ff} + l_r u_c$
$u = u + u_{ff}$
Output u
Update State:
$\hat{x} = \Phi\hat{x} + \Gamma u + K(y - C\hat{x})$
$x_m = \Phi x_m + \Gamma u_{ff}$
$u_{ff} = -L_m x_m$
$u = L(x_m - \hat{x})$

5.

a. Since $1 \leq a < 2$, only one bit is needed to represent the integer part and $8-1-1 = 6$ bits are left for the fractional part. The fixed point representation of a is $\text{round}(1.35 \cdot 2^6) = 86$.

b. In the following code, the result of the multiplication is saved in a temporary variable, which has type `int16_t`. The value is then saturated to handle overflows and underflows and finally casted into its corresponding `int8_t` value.

```
#include <inttypes.h>
// Insert in the next two lines the
// results from the first subproblem
#define n 6
#define a 86

int8_t x, b;
// define more variables if needed
int16_t temporary;

// assume b is initialized to a value (you don't need to write
// the code for the initialization of b)
// write the code to compute x
temporary = ((int16_t) a*b) >> n;
if (temporary > 127)
  temporary = 127;
else if (temporary < -128)
  temporary = -128;

x = (int8_t) temporary;
```

6.

a. Denoting the control task execution time with x it is possible to compute the utilization as

$$U = \frac{x}{4} + \frac{4}{12} + \frac{9}{20}$$

where 20 is the period of the user interaction tasks in milliseconds obtained as 1000/50. For schedulability the utilization should be less than 1 and imposing the conditions above gives $x < 0.867$ milliseconds.

b. As stated above, the task set is schedulable using EDF. For monotonic priority assignments, a sufficient schedulability criterion is if the utilization satisfies the condition

$$\sum_{i=1}^{i=n} \leq n(2^{1/n} - 1).$$

But the utilization for the given task set is

$$\sum_{i=1}^{i=n} = 0.5/4 + 4/12 + 9/20 = 0.9083 \geq 3(2^{1/3} - 1) \approx 0.78$$

and we can not conclude that the tasks are schedulable. We instead turn to the exact analysis, where the response time for process i can be calculated from iteration of the equation

$$R_i = C_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j.$$

The highest priority task is the control task, which will trivially have a response time which we denote $R_1 = C_1 = x = 0.5$ ms. For the medium priority reference generator we get, with initial guess $R_2^0 = C_2 = 4$:

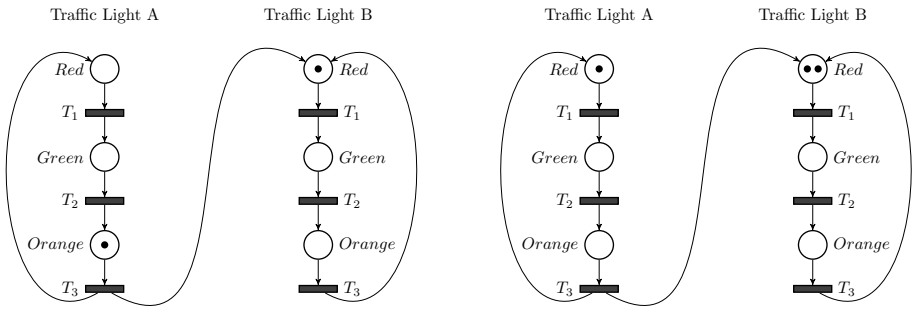
$$R_2^1 = C_2 + \left\lceil \frac{R_2^0}{T_1} \right\rceil C_1 = 4 + \left\lceil \frac{4}{4} \right\rceil 0.5 = 4.5$$
$$R_2^2 = C_2 + \left\lceil \frac{R_2^1}{T_1} \right\rceil C_1 = 4 + \left\lceil \frac{4.5}{4} \right\rceil 0.5 = 5.0$$
$$R_2^3 = C_2 + \left\lceil \frac{R_2^2}{T_1} \right\rceil C_1 = 4 + \left\lceil \frac{5.0}{4} \right\rceil 0.5 = 5.0$$

which gives $R_2 = 5.0$ ms. For the user I/O task, with initial guess $R_3^0 = C_3 = 9$ we get

$$R_3^1 = C_3 + \left\lceil \frac{R_3^0}{T_1} \right\rceil C_1 + \left\lceil \frac{R_3^0}{T_2} \right\rceil C_2 = 9 + \left\lceil \frac{9}{4} \right\rceil 0.5 + \left\lceil \frac{9}{12} \right\rceil 4 = 14.5$$
$$R_3^2 = C_3 + \left\lceil \frac{R_3^1}{T_1} \right\rceil C_1 + \left\lceil \frac{R_3^1}{T_2} \right\rceil C_2 = 9 + \left\lceil \frac{14.5}{4} \right\rceil 0.5 + \left\lceil \frac{14.5}{12} \right\rceil 4 = 19$$
$$R_3^3 = C_3 + \left\lceil \frac{R_3^2}{T_1} \right\rceil C_1 + \left\lceil \frac{R_3^2}{T_2} \right\rceil C_2 = 9 + \left\lceil \frac{19}{4} \right\rceil 0.5 + \left\lceil \frac{19}{12} \right\rceil 4 = 19.5$$
$$R_3^4 = C_3 + \left\lceil \frac{R_3^3}{T_1} \right\rceil C_1 + \left\lceil \frac{R_3^3}{T_2} \right\rceil C_2 = 9 + \left\lceil \frac{19.5}{4} \right\rceil 0.5 + \left\lceil \frac{19.5}{12} \right\rceil 4 = 19.5$$

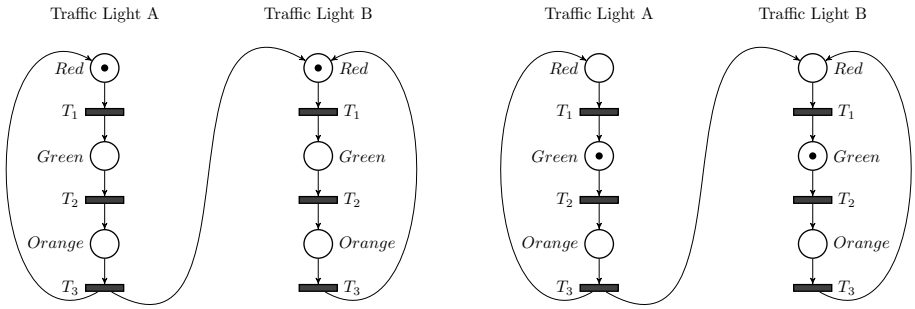
which gives $R_3 = 19.5$ ms. All tasks will therefore meet their deadlines.

7.



(a) Before A.T3 firing (b) After A.T3 firing

Figure 1 Illustration of the unboundedness of the suggested solution.



(a) Before A.T1 and B.T1 firing (b) After A.T1 and B.T1 firing

Figure 2 Illustration of the lack of mutual exclusion of the suggested solution.

a. The suggested solution is indeed unbounded. Every time traffic light A fire transition T3 a new token will appear in B.Red, even though it might already have one there. See Fig. 1.

The suggested solution is clearly not mutually exclusive since there is nothing preventing it from firing both A.T1 and B.T1 simultaneously, see Fig 2.

b. To solve all problems all one needs to do is to add a semaphore that is shared between the two traffic lights. The extended solution can be seen in Fig. 3.

8 a. The output will be

```
Output 1 = 7
Output 2 = 3
Output 3 = 9
```

b. `a` is a local variable which is assigned the value `7` in the main method. The fact that `a` is passed as an argument to `myMethod` will not change its value, since Java uses call-by-value when simple data types are used as method arguments. `ref1` is a reference to a `MyClass` object, which is then passed as an argument to the method `myMethod`. Since Java uses call-by-reference

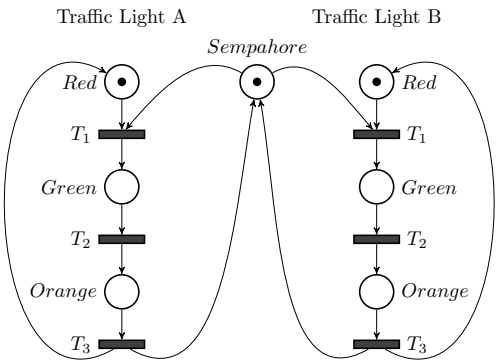


Figure 3 Solution that ensures both boundedness and mutual exclusion between the green lights.

when passing objects as arguments to methods the assignment `obj.a = 3` will be performed on the object referenced by `ref1` (actually Java uses call-by value also in this case since it is a reference that is the argument to the method). Since `b` is declared as `static` all instances of `MyClass` will share this attribute. Hence, the assignment `obj.b = 9` in `myMethod` will effect also the object referenced by `ref2`.