Tentamen 7 April 2016

Real-time Systems (Technische Universiteit Delft)

TU Delft

TECHNISCHE UNIVERSITEIT DELFT
Faculteit Elektrotechniek, Wiskunde en Informatica

IN4343 – Real Time Systems

April 7th 2016, from 09:00 to 12:00

Koen Langendoen

Question:	1	2	3	4	5	6	Total
Points:	10	5	20	15	25	15	90
Score:							

- This is a closed book exam
- You may use a **simple** calculator only (i.e. graphical calculators are not permitted)
- Write your answers with a black or blue pen, not with a pencil
- **Always justify your answers, unless stated otherwise**

- The exam covers the following material:
  (a) chapters 1-6, 8-9 of the book "Hard Real-Time Computing Systems (3rd ed)" by G. Buttazzo
  (b) the paper "The Worst-Case Execution-Time Problem" by Wilhelm et al. (except Section 6)
  (c) the paper "Transforming Execution-Time Boundable Code into Temporally Predictable Code" by P. Puschner
  (d) the paper "Best-case response times and jitter analysis of real-time tasks" by R.J. Bril, E.F.M. Steffens, and W.F.J. Verhaegh

Liu and Layland (LL) bound		$U_{lub}^{RM} = n(2^{1/n} - 1)$
Hyperbolic (HB) bound		$\prod_{i=1}^{n}(U_i + 1) \leq 2$
Response Time Analysis		$WR_i = C_i + \sum_{k=1}^{i-1} \left\lceil \frac{WR_i + AJ_k}{T_k} \right\rceil C_k$ $BR_i = C_i + \sum_{k=1}^{i-1} \left(\left\lceil \frac{BR_i - AJ_k}{T_k} \right\rceil - 1 \right)^+ C_k$ $w^+ = \max(w, 0)$
Processor Demand		$g(t_1, t_2) = \sum_{r_i \geq t_1}^{d_i \leq t_2} C_i \qquad g(0, L) = \sum_{i=1}^{n} \left\lfloor \frac{L + T_i - D_i}{T_i} \right\rfloor C_i$
	schedulability	$\forall L \in D, \quad g(0, L) \leq L$ $D = \{d_k \mid d_k \leq \min(H, \max(D_{max}, L^*))\}$ $H = lcm(T_1, \ldots, T_n)$ $L^* = \frac{\sum_{i=1}^{n}(T_i - D_i)U_i}{1 - U}$
Polling Server	schedulability	$U_{lub}^{RM+PS} = U_s + n\left[\left(\frac{2}{U_s + 1}\right)^{1/n} - 1\right]$ $\prod_{i=1}^{n}(U_i + 1) \leq \frac{2}{U_s + 1}$
	response time	$R_a = C_a + \Delta_a + F_a(T_s - C_s)$ $\Delta_a = \left\lceil \frac{r_a}{T_s} \right\rceil T_s - r_a \qquad F_a = \left\lceil \frac{C_a}{C_s} \right\rceil - 1$
Deferrable Server	schedulability	$U_{lub}^{RM+DS} = U_s + n\left[\left(\frac{U_s + 2}{2U_s + 1}\right)^{1/n} - 1\right]$ $\prod_{i=1}^{n}(U_i + 1) \leq \frac{U_s + 2}{2U_s + 1}$
	response time	$R_a = C_a + \Delta_a - C_0 + F_a(T_s - C_s)$ $C_0 = \min(C_s(r_a), \Delta_a)$ $\Delta_a = \left\lceil \frac{r_a}{T_s} \right\rceil T_s - r_a \qquad F_a = \left\lceil \frac{C_a - C_0}{C_s} \right\rceil - 1$
NP scheduling	level-i busy period	$L_i = B_i + \sum_{h=1}^{i} \left\lceil \frac{L_i}{T_h} \right\rceil C_h \qquad N_i = \left\lceil \frac{L_i}{T_i} \right\rceil$ $B_i = \max_{j>i}\{C_j\}$
	response time	$s_{ik} = B_i + (k-1)C_i + \sum_{h=1}^{i-1} \left(\left\lfloor \frac{s_{ik}}{T_h} \right\rfloor + 1 \right) C_h$ $R_{ik} = (s_{ik} + C_i) - (k-1)T_i$ $R_i = \max_{k \in [1, N_i]}\{R_{ik}\}$
Elastic Model	utilization	$\forall i \quad U_i = U_{i0} - (U_0 - U_d)\frac{E_i}{E_S} \qquad$ where $E_S = \sum_{i=1}^{n} E_i$

Question 1 [10 points]

Determining the worst-case execution time of a task lies at the heart of real-time systems.

(a) [2 points] Provide two reasons explaining the variability observed in practice when measuring the execution time of a task.

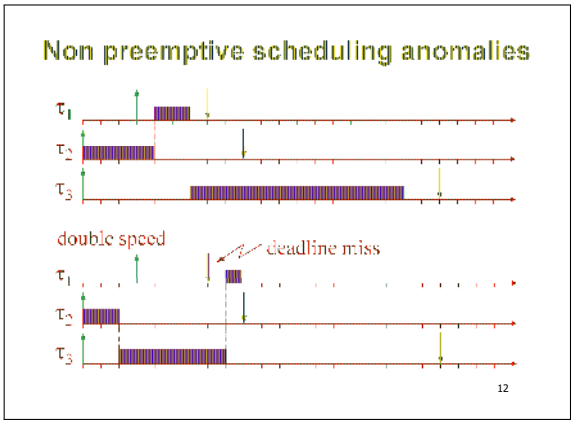
Solution: (Wilhelm:2008) mentions three reasons: Data-Dependent Control Flow, Context Dependence of Execution Times, and Timing Anomalies.

(b) [3 points] Explain why **dynamic** timing analysis rarely determines the true WCET of a task.

Solution: Assuming that only a fixed amount of time is available, one can only measure a fixed number of possible executions (e.g., setting of task parameters). Hence, for all but the most trivial codes it is highly unlikely that the longest execution is amongst the set of measured ones.

(c) [5 points] Explain the concept of scheduling anomalies, and give an example in which knowing the exact WCETs still leads to a deadline violation.

Solution: The problem is that for **non-preemptive** scheduling an undershoot may lead to a different execution order causing a deadline violation. An example from lecture 13:



Question 2 [5 points]

Response Time Analysis (RTA) is the key technique to analyze Rate Monotonic scheduling. In the simplest case, when looking at the worst-case response times for tasks without jitter, the RTA equation boils down to:

$$R_i = C_i + \sum_{k=1}^{i-1} \left\lceil \frac{R_i}{T_k} \right\rceil C_k$$

(a) [2 points] Argue why it is impossible to derive a closed-form (non-recursive) equation for R_i.

Solution: The problem is the ceiling operator in the recursive equation, which prevents left factoring the R_i expression on the right hand side.

(b) [1 point] The RTA equation can be solved by an iterative procedure. Although starting with $R_i^0 = C_i$ will find the lowest fixed-point solution, one can do better (i.e. fewer iterations) by starting at a higher value. Show that

$$R_i^* = C_i + \sum_{k=1}^{i-1} \frac{R_i^*}{T_k} C_k$$

is a valid starting point (i.e. $R_i^* \leq R_i$).

Solution: $R_i^* = C_i + \sum_{k=1}^{i-1} \left(\frac{R_i^*}{T_k} \right) C_k \leq C_i + \sum_{k=1}^{i-1} \left\lceil \frac{R_i^*}{T_k} \right\rceil C_k = R_i$

(c) [2 points] Provide a closed-form expression for R_i^*.

Solution: $R_i^* = C_i + \sum_{k=1}^{i-1} \left(\frac{R_i^*}{T_k} \right) C_k = C_i + R_i^* \sum_{k=1}^{i-1} \frac{C_k}{T_k} \implies R_i^* \left(1 - \sum_{k=1}^{i-1} \frac{C_k}{T_k} \right) = C_i \implies$

$$R_i^* = \frac{C_i}{1 - \sum_{k=1}^{i-1} \frac{C_k}{T_k}}$$

Question 3 [20 points]

	C_i	T_i
τ_1	2	5
τ_2	2	9
τ_3	6	24
τ_4	4	48

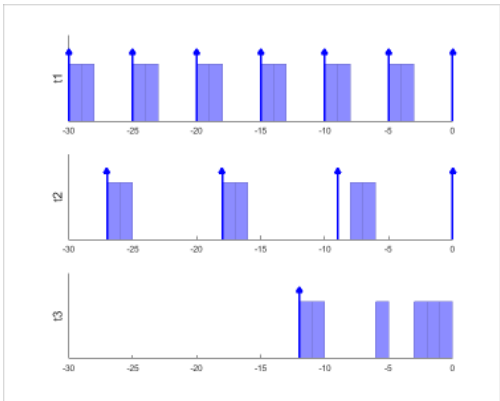
(a) [10 points] Compute the worst-case response times of the tasks under RM scheduling.

Solution: You can/should use the starting values from the previous question.

	R_i^*	R_i
τ_1	2	2
τ_2	4 (3.3)	4
τ_3	16 (15.9)	18
τ_4	32 (31.3)	44

(b) [5 points] Draw the optimal instant for task τ_3, and report its best-case response time.

Solution: The best-case response time is 12.



(c) 2 points Explain why one usually **cannot** derive the exact reponse jitter when the worst- and best-case response time of a task are known.

Solution: The catch is that the respective response times will usually be achieved for two different phasing of the tasks involved.

(d) 3 points Compute the bounds on the response jitter of the tasks.

Solution:

	RJ_i
τ_1	0
τ_2	2
τ_3	6
τ_4	38

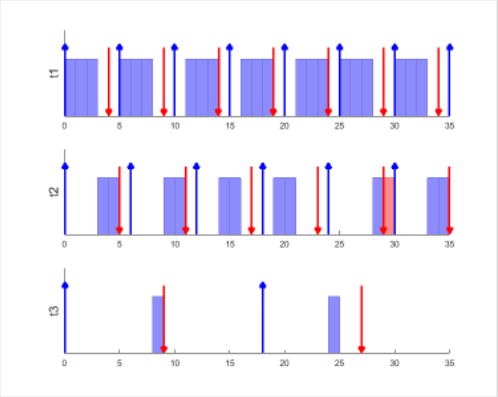
Question 4 [15 points]

	C_i	D_i	T_i
τ_1	3	4	5
τ_2	2	5	6
τ_3	1	9	18

(a) [10 points] Determine if the task set is feasible under EDF scheduling.

Solution: Processor utilization is $89/90 \leq 1$, so we need to check the processor demand until $min(H, max(D_{max}, L^*))$ with $H = 90$, $D_{max} = 9$, and $L^* = 129$, so until 90.

	demand
g(0,4)	3
g(0,5)	5
g(0,9)	9
g(0,11)	11
g(0,14)	14
g(0,17)	16
g(0,19)	19
g(0,23)	21
g(0,24)	24
g(0,27)	25
g(0,29)	30

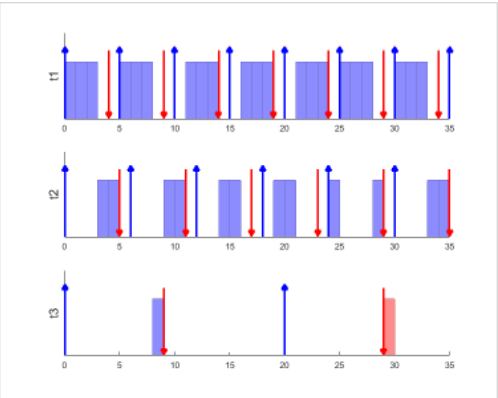


Task set is **un**feasible!

(b) [5 points] Determine if the task set is (still?) feasible under EDF scheduling when relaxing the period of task τ_3 to 20.

Solution: Processor utilization is $59/60 \leq 1$, so we need to check the processor demand until $min(H, max(D_{max}, L^*))$ with $H = 60$, $D_{max} = 9$, and $L^* = 89$, so until 60.
Note that everything stays the same until the second deadline of task τ_3 ($t = 29$ vs. $t = 27$).

	demand
g(0,24)	24
g(0,29)	30



Task set is **un**feasible, but now task τ_3 misses its deadline!

Question 5 [25 points]

Although preemptive scheduling is the norm, non-preemptive scheduling has certain advantages that give it an edge in specific conditions. However, it complicates matters as well, reducing schedulability and making analysis more difficult.

Consider the following task set:

	C_i	T_i
τ_1	3	9
τ_2	4	12
τ_3	4	15

(a) 2 points Name two advantages of non-preemptive scheduling.

Solution: NP scheduling (i) reduces context-switching overhead, (ii) reduces (nulls) jitter, (iii) simplifies access to shared resources, and (iv) simplifies stack management.

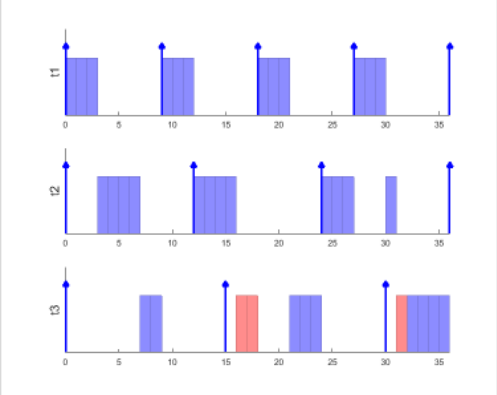
(b) 3 points Explain why, in the case of RM scheduling, it is no longer sufficient to analyze the critical instant when determining the feasibility of a task set.

Solution: That is due to the **self-pushing phenomenon**. High-priority tasks activated during execution of lower-priority tasks are pushed ahead and introduce extra delays in subsequent jobs of the same task.

(c) 5 points Mention the two conditions under which it suffices to consider only one scheduling period to derive the worst-case response times, and show if they hold for the given task set.

Solution:
1. The task set is feasible under preemptive scheduling (FALSE, see below)
2. All deadlines are less than or equal to the periods (TRUE)

Util ≤ 1 and HB bound fails, so revert to RTA / timeline:



(d) 3 points Explain the notion of **level-i busy period** and how it can be used to determine the maximum number of periods to derive the worst-case response time of a task.

Solution: The level-i-busy period captures the interval in which the processor is busy executing tasks with priority higher than or equal to task i including blocking times.

(e) 12 points Determine the feasibility of the task set under non-preemptive RM scheduling by reporting (i) the level-i busy periods, and (ii) the worst-case response times of the tasks.

Solution:

	B_i	L_i	N_i	R_i
τ_1	4	**7**	1	**7**
τ_2	4	**18**	2	**11**
τ_3	0	**36**	3	**11**

Task set is feasible.

Question 6 [15 points]

Consider the set of tasks below:

	C_i	T_i^{min}	T_i^{max}	E_i
τ_1	5	8	15	1
τ_2	6	10	20	2
τ_3	7	12	30	2

(a) 3 points Explain the difference between a **transient** and **permanent** overloaded system.

Solution: For a transient (temporarily) overloaded system the average load (utilization) is below 1. In contrast a permanent overload implies the average load exceeds 1.

(b) 4 points The Elastic Period Method can be used to derive the periods T_i that would guarantee the schedulability of the given tasks under a specific scheduling policy.

– Explain the complication of "boundary conditions", and how to handle them.

Solution: The problem is that the computed periods T_i should be within the respective intervals $[T_i^{min}, T_i^{max}]$. If the EP method yields a period that is outside the interval of some task then that task should be frozen at the boundary, and the computation must be run again until all of the tasks are within their bounds.

(c) 8 points Apply the Elastic Period Method to the above task set for EDF and compute the periods T_i. Assume that all tasks start at their maximum rate (i.e. with T_{min}).

Solution: For EDF we know that the utilization (U_d) has to be ≤ 1. Using the elastic method we get:

$U_S = \frac{5}{8} + \frac{6}{10} + \frac{7}{12} = \frac{75+72+70}{120} = \frac{217}{120}$

$U_1 = \frac{5}{8} - (U_S - 1)\frac{1}{5} = 0.463, \quad T_1 = \frac{5}{U_1} = 10.79$

$U_2 = \frac{6}{10} - (U_S - 1)\frac{2}{5} = 0.277, \quad T_2 = \frac{6}{U_2} = 21.69$

Task 2 exceeds the maximum interval so it needs to be fixed at $T_2 = 20$. Recompute for tasks 1 and 3:

$U_S = \frac{5}{8} + \frac{6}{20} + \frac{7}{12} = \frac{75+36+70}{120} = \frac{181}{120}$

$U_1 = \frac{5}{8} - (U_S - 1)\frac{1}{3} = 0.456, \quad T_1 = \frac{5}{U_1} = 10.98$

$U_3 = \frac{7}{12} - (U_S - 1)\frac{2}{3} = 0.244, \quad T_3 = \frac{7}{U_3} = 28.63$

Note: the original exam contained a small mistake (T_3^{max} was set to 25) forcing an extra iteration. Students who did so got a bonus of max 4 pts.