# Tentamen 2020 Version 1

Real-time Systems (Technische Universiteit Delft)

Delft University of Technology
Faculty of Electrical Engineering, Mathematics, and Computer Science (EEMCS)

# IN4343 – Real-Time Systems
# Final Exam 2019/2020

Instructor: Mitra Nasri
Duration: **3 hours and 15 minutes** (**9:00 to 12:15**)

| Question: | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Total |
|---|---|---|---|---|---|---|---|---|
| Points: | 15 | 5 | 10 | 10 | 10 | 15 | 5 | 70 |
| Score: | | | | | | | | |

## Code of honor

"I promise that I have not used unauthorized help from people or other sources for completing my exam. I created the submitted answers all by myself during the time slot that was allocated for that specific exam part."

## Important definitions

- **Main Exam:** it refers to the exam that is set up as a Brightspace (BS) quiz called **Main-Exam-15-4-2020**.
- **ExamUploadBox :** it refers to the BS assignment called **Exam-Uploadbox-15-4-2020**.
- **Answer paper:** it is a paper that starts with your name and student ID. You can answer only 1 question on each answer paper. You must write the number of question you answer on a paper on top of the paper. **Answer papers without your name and ID will not be taken into account for grading.** The file of an answer paper must be named with the question number, for example: **Q1.jpeg** or **Q1.pdf**.
- **Draft paper:** these are your draft papers. During your online interview, you can only refer to these pages to justify that you have reached to the solution by yourself. Your name and ID must be on top of each draft paper. You must upload the picture of all of your draft papers on ExamUploadBox . Name draft files as **Draft1.jpeg** or **Draft1.pdf**.

## Instructions for the online exam

(a) The exam is closed book. You are not allowed to ask help from anyone or open any note/book/slide apart from the exam.

(b) You must be alone in a quite room during the exam.

(c) Do not connect to social networks on your mobile device or laptop during the exam (unless you want to ask a question from the instructor). Our **Mattermost forum** is considered as a social network and is **forbidden** to use during the exam.

(d) **Do not disclose your answers or questions** to anyone during or **after the exam**.

(e) **Do not disclose the contents of your oral interview or that you were interviewed to anyone until the instructor announces the end of examination.**

(f) **Upload your answer papers and draft papers only on the ExamUploadBox** . None of the files that have been received via other means (such as email) will be graded.

(g) If it has not been mentioned specifically, you must type your answers on MainExam . Only very few questions ask you to upload your answers on ExamUploadBox .

(h) If you take a picture of your answer paper, make sure that the uploaded file has the right angle (vertical) and is readable without any issue.

(i) If you have an issue with Brightspace or uploading files, you can contact the instructor via email: Mitra Nasri m.nasri@tudelft.nl

## Instructions for the exam

- You may use a **simple** calculator only (it is fine if you use your device/laptop calculator as long as it is simple and cannot store equations).
- **Always justify your answers**. Without justifying your answer, you may not get a point unless stated otherwise.
- EDF and FP refer to the **preemptive** Earliest-Deadline-First and **preemptive** Fixed-Priority scheduling algorithms unless it is explicitly mentioned that they are non-preemptive.
- Assume that there is no preemption overhead unless it is stated explicitly in the question.
- Assume that a smaller priority indicates a higher priority for the FP scheduling algorithm.
- The exam covers the following material:
  (a) chapters 1-9 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 "Non-Work-Conserving Non-preemptive Scheduling: Motivations, Challenges, and Potential Solutions" by Mitra Nasri and Gerhard Fohler
  (d) the paper "An Exact and Sustainable Analysis of Non-Preemptive Scheduling" by Mitra Nasri and Björn B. Brandenburg

## IN4343 Real-Time Systems: Cheat Sheet

| | |
|---|---|
| **Liu and Layland (LL) test** | (1) $U \leq n(2^{1/n} - 1)$ |
| **Utilization-bound for partitioned EDF with First-Fit** | (2) $U^{EDF+FF} = (m+1)/2$ |
| **Response-Time Analysis (FP) No jitter:** | (3) $R_i = C_i + \sum_{k=1}^{i-1} \left\lceil \frac{R_i}{T_k} \right\rceil C_k$<br>(4) $R_i^{(0)} = C_i$ |
| **Demand Bound Analysis**<br><br><br>**Schedulability** | (5) $g(t_1, t_2) = \sum_{r_i \geq t_1}^{d_i \leq t_2} C_i$<br>(6) $g(0, L) = \sum_{i=1}^{n} \left\lfloor \frac{L + T_i - D_i}{T_i} \right\rfloor C_i$<br>(7) $\forall L \in \overline{D}, \quad g(0, L) \leq L$<br>(8) $\overline{D} = \{d_k \| d_k \leq \min(H, \max(D_{max}, L^*))\}$<br>(9) $H = lcm(T_1, \ldots, T_n)$<br>(10) $D_{max} = \max\{D_i \mid \tau_i \in \tau\}$<br>(11) $L^* = \frac{\sum_{i=1}^{n}(T_i - D_i) \cdot U_i}{1 - U}$ |
| **CBS Server: Arrival of a job $J_k$**<br><br><br>**Budget exhaustion** | **if** ($\exists$ a pending aperiodic job) **then** enqueue $J_k$;<br>**else if** $(r_k + (q_s/U_s) > d_s)$ **then**<br>$\{ q_s \leftarrow Q_s; \quad d_s \leftarrow r_k + T_s; \}$<br>$\{ q_s \leftarrow Q_s; \quad d_s \leftarrow d_s + T_s \}$ |
| **Response-time NP-FP**<br><br><br><br><br><br>**Schedulability test NP-EDF** | pre-requisites: $D \leq T$ and preemptive-schedulable<br>(12) $WO_i^{(n)} = B_i + \sum_{k=1}^{i-1} \left( \left\lfloor \frac{WO_i^{(n-1)}}{T_k} \right\rfloor + 1 \right) \cdot C_k$<br>(13) $B_i = \max\{C_j \mid \forall \tau_j, P_i < P_j\}$<br>(14) $R_i = C_i + WO_i$<br>(15) $WO_i^{(0)} = B_i + \sum_{k=1}^{i-1} C_k$<br>$\forall \tau_i, 1 < i \leq n, \forall L, T_1 < L < T_i$:<br>(16) $L \geq C_i + \sum_{k=1}^{i-1} \left\lfloor \frac{L-1}{T_k} \right\rfloor \cdot C_k$ |

## Question 1 **[15 points]**

Consider a uniprocessor system with a set of sporadic tasks scheduled with a fixed-priority scheduling policy. The task set has $n$ tasks $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$ that are sorted and indexed by priority, namely, $P_1 < P_2 < \ldots < P_n$ (i.e., task $\tau_1$ has the highest and task $\tau_n$ has the lowest priority). **All of these tasks are fully preemptive except task $\tau_k$ ($1 < k < n$ and $k$ is given).** Assume that the task set is schedulable if all tasks were fully preemptive.

(a) $\boxed{\text{10 points}}$ **Q1-1.** Write down equations to calculate a safe upper bound on the worst-case response time (WCRT) of a task $\tau_i$, where $1 \leq i < k$, i.e., task $\tau_i$ has a higher priority than task $\tau_k$.

Briefly justify the correctness of your equations. Note: you are free to refer to the equation numbers in the cheatsheet if you believe that the equation can be directly used for task $\tau_i$.

**Instructions:** Upload your equations in ExamUploadBox (with file name "Q1-1") and write down your justifications in MainExam (in the box provided in this question).

**Solution:** Tasks that have higher priority than $\tau_k$ can be blocked by $\tau_k$ for at most $C_k$ units of time. So, for those tasks, we need to consider a blocking factor

$$B_i = C_k$$

Since each of these tasks such as $\tau_i$: (i) is preemptive and (ii) can be blocked only once by the low priority task $\tau_k$, its worst-case response time will be not be larger than its worst-case response time under the preemptive scheduling plus the blocking time caused by $\tau_k$. Hence, we have

$$R_i^{(0)} = B_i + \sum_{j=1}^{i} C_j$$

$$R_i^{(p)} = B_i + C_i + \sum_{j=1}^{i-1} \left\lceil \frac{R_i^{(p-1)}}{T_j} \right\rceil \cdot C_j$$

Note that any of the following $R_i^{(0)}$ equations is also acceptable: $R_i^{(0)} = B_i + C_i$, or $R_i^{(0)} = B_i + C_i + R_{i-1}$

**Grading Rubric:** Solutions that do not incorporate the blocking factor in the WCRT are wrong. **1 point** for a correct $R_i^{(0)}$ (initial value). **1 point** for recognizing that the preemptive equations should be used with fixed-point iteration. Solutions without fixed-point iteration will receive 0 points. **8 points** for correct identification of blocking (taking the blocking factor $C_k$ into account) with correct equation. If only the blocking is identified correctly but there is a mistake in equation, only 2 points will be given. If the blocking has not been incorporated into the equation, then 0 points out of 8 is given here.

(b) $\boxed{\text{5 points}}$ **Q1-2.** Write down equations to calculate a safe upper bound on the WCRT of the task $\tau_k$ (the non-preemptive task).

Briefly justify the correctness of your equations. Note: you are free to refer to the equation numbers in the cheatsheet if you believe that the equation can be directly used for task $\tau_k$.

**Instructions:** Upload your equations in ExamUploadBox (with file name "Q1-2") and write down your justifications in MainExam (in the box provided in this question).

**Solution:**

Since the scheduling algorithm is fixed-priority and since all of the lower-priority tasks than $\tau_k$ are preemptive, they cannot block $\tau_k$, namely:

$$B_k = 0$$

Task $\tau_k$ can start its execution only if all higher-priority tasks have been completed, hence, its start time follows equation (12), namely:

$$WO_k^{(0)} = \sum_{j=1}^{k-1} C_j$$

$$WO_k^{(p)} = \sum_{j=1}^{k-1} \left( \left\lfloor \frac{WO_k^{(p-1)}}{T_j} \right\rfloor + 1 \right) \cdot C_j$$

$$R_k = C_k + WO_k$$

Note that the followings are also correct: $WO_k^{(0)} = 0$ or $WO_k^{(0)} = R_{k-1}$

It is important to note that even though $\tau_k$ cannot be blocked by other tasks, its WCRT does not follow equation (3) because it cannot be preempted by other tasks. Equation (3) accounts for the preemptions of the task under analysis because it integrates them in the fixed-point iteration by adding the execution time of the task itself to the response time equation.

**Grading Rubric:**

- Solutions that incorporate the blocking factor in the WCRT are wrong and will not receive points. Solutions that do not separate the start time from the response time and include $C_k$ in the fixed-point iteration will receive **0 points**. **3 points** for using Equation (12). **1 point** for using equation (14) for the response time. **1 point** a correct $WO_k^{(0)}$ (initial value).
- **Exceptions: 3 points** for solutions that are based on $R_{k-1} + C_k$ and introduce the $R_{k-1}$ with preemptive equations. If $R_{k-1}$ has not been introduced or is obtained by wrong equation, then only **2 points** will be given.

## Question 2 [5 points]

(a) $\boxed{\text{5 points}}$ **Q2-1.** The following task set is scheduled by EDF.

| | $C_i$ | $T_i$ | $D_i$ |
|---|---|---|---|
| $\tau_1$ | 3 | 8 | 5 |
| $\tau_2$ | 10 | 20 | 16 |
| $\tau_3$ | 6 | 90 | 87 |

List the time instants that "**must**" be evaluated in the demand-bound analysis (DBA) for EDF assuming that the fast DBA schedulability test is applied (the one that uses $L^*$).

**Solution:**

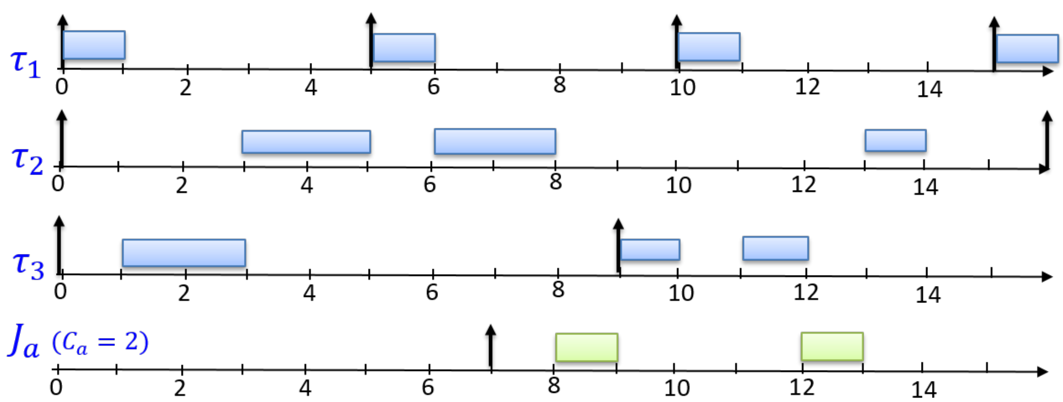
$$L^* = \frac{\sum_{i=1}^{n}(T_i - D_i)\cdot U_i}{1-U} = 57$$

$$\begin{aligned} D &= \{d_k | d_k \leq \min(H, \max(D_{max}, L^*))\} \\ &= \{d_k | d_k \leq \min(LCM(8, 20, 90), \max(87, 57))\} \\ &= \{d_k | d_k \leq 87\} \\ &= \tau_1 : \{5, 13, 21, 29, 37, 45, 53, 61, 69, 77, 85\} \cup \tau_2 : \{16, 36, 56, 76\} \cup \tau_3 : \{87\} \end{aligned}$$

**Grading Rubric:** **1 point** if the values are up to 87. **2 points** if the number patterns are correct. **2 points** for for correct L* = 57.

## Question 3 [10 points]

For the next sub-questions, consider the following schedule generated by fixed-priority scheduling for a set of preemptive periodic tasks $\tau = \{\tau_1, \tau_2, \tau_3\}$) and an aperiodic job $J_a$ which has been scheduled inside a server. The first job of the server has been released at time 0 and the period of the server is an integer value $\leq 12$.

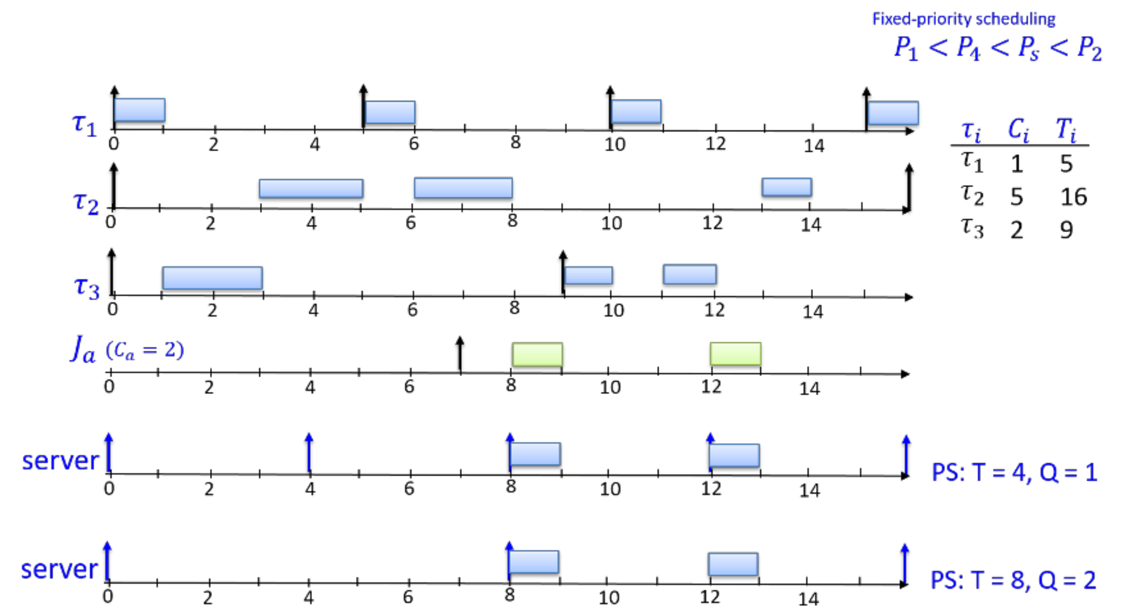



| $\tau_i$ | $C_i$ | $T_i$ |
|---|---|---|
| $\tau_1$ | 1 | 5 |
| $\tau_2$ | 5 | 16 |
| $\tau_3$ | 2 | 9 |

(a) $\boxed{\text{3 points}}$ **Q3-1.** What are the **budget** [Blank1] , **period** [Blank2] and **type** [Blank3] of the reservation server?

**Important assumptions:** (1) server type can only be: {"PS", "DS", "periodic server"}, (2) period of the server must be an integer ≤ 12, and (3) the combination of the server and the three periodic tasks must be schedulable by fixed-priority scheduling.

**Solution:** It is a polling server (PS) with period 4 and budget 1.




**Grading Rubric: 3 points** if the answer is correct and **0 points** if there is any mistake.

(b) $\boxed{\text{3 points}}$ **Q3-2.** What are the priority relations between the tasks and the server? Justify your answer. You should refer to the schedule. **Answers *without justification* will not be taken into account even if they are correct.**

**Solution:** Priority relation: $P_1 < P_3 < P_s < P_2$ because

(a) $P_1 <$ **others**: We see that task $\tau_1$ has been scheduled always immediately after its arrival regardless of the presence of the aperiodic job.

(b) $P_1 < P_3 < \{P_2, P_s\}$: Since the first job of $\tau_3$ has been scheduled immediately after the first job of $\tau_1$, $P_1 < P_3$. Moreover, since during the interval [8, 14], $\tau_2$ and the server had to wait for $\tau_3$, the priority of $\tau_3$ must have been higher than the two.

(c) $P_1 < P_3 < P_s < P_2$: during intervals [8, 9] and [12, 13], the server was running while $\tau_2$ was pending (preempted), hence the priority of $\tau_2$ must be lower than the server.

**Grading Rubric: 1 point** for each of the above items.

(c) $\boxed{\text{2 points}}$ **Q3-3.** Explain why the server could not be of any other type than what you claim?

**Solution:**

(a) **The server cannot be DS**: The job $J_a$ has been released at time 7, where all of the higher-priority tasks have been completed. If the server was DS, then it must have had at least 1 unit of budget that could be used to schedule $J_a$ in [7, 8] (note that the first release of the server was at time 0).
This means that the server had no budget at time 7. This could have happened only

if the server was PS or periodic server.

(b) **The server cannot be a periodic server**: because before the release time of the aperiodic job, the server has not been scheduled.

**Grading Rubric:** **1 point** for each correct justification from the above list.

(d) 2 points **Q3-4.** Explain how your proposed server will result in this schedule. You are allowed to upload a figure or drawing in ExamUploadBox if you think it helps explaining the answer. However, the main answer must be entered here in MainExam .

**Solution:** The figure in the solution of part (a) shows the execution of the server.
**Justifying execution of $J_a$ during interval [8, 9] and [12, 13]**: since the server has been activated at time 8, and since $J_a$ has been released prior to 8, the budget can be used to schedule $J_a$. But then at time 9, the budget finishes. Now $J_a$ must wait until the next activation of the server at time 12.

**Grading Rubric:** **2 points** for a correct answer. 0 point for an answer which has a mistake or has not explained the execution of $J_a$ in the intervals [8, 9] and [12, 13].

## Question 4 [10 points]

For the next two sub-questions, use the task set below:

| Task | $\tau_1$ | $\tau_2$ | $\tau_3$ | $\tau_4$ | $\tau_5$ | $\tau_6$ | $\tau_7$ | $\tau_8$ |
|---|---|---|---|---|---|---|---|---|
| WCET | 9 | 16 | 1 | 5 | 4 | 6 | 3 | 1 |
| Minimum inter-arrival time | 10 | 20 | 2 | 10 | 20 | 20 | 10 | 10 |

(a) 3 points **Q4-1. Without partitioning** the task set, estimate what is the minimum number of cores needed to ensure that the task set is certainly schedulable with partitioned EDF using a **first-fit** partitioning heuristic?

**Solution:**
Tasks have the following utilizations: U1 = 0.9, U2 = 0.8, U3 = 0.5, U4 = 0.5, U5 = 0.2, U6 = 0.3, U7 = 0.3, U8 = 0.1.
The total utilization is $U^{tot} = \sum_{i=1}^{8} U_i = 3.6$
The minimum number of cores with partitioned EDF + FF is given by $m$, where $U^{tot} \leq \frac{m+1}{2}$. Thus, we get $m \geq 2 \times 3.6 - 1 = 6.2$. It results that the minimum number of cores to ensure that the task set is schedulable with partitioned EDF+FF is $m = 7$.

**Grading Rubric:** **3 points** for a correct answer and correct justification. **0 points** if there is any mistake.

(b) 7 points **Q4-2.** What is the worst-case response time of task $\tau_6$ if: (1) the task set is partitioned using **best-fit** considering tasks in a decreasing task utilization order and using $U^{tot} \leq 1$ as a fitting function, and (2) tasks are scheduled with rate-monotonic on each core.
**Explain each and every step to reach to your answer.**

**Solution:**
Best-fit with decreasing utilization partitions task in the following order: $\tau_1$, $\tau_2$, $\tau_3$, $\tau_4$, $\tau_6$, $\tau_7$, $\tau_5$, $\tau_8$. The resulting partitioning with $U^{tot} \leq 1$ is
core 1: $\tau_1$, $\tau_8$;
core 2: $\tau_2$, $\tau_5$;
core 3: $\tau_3$, $\tau_4$;
core 4: $\tau_6$, $\tau_7$.
Thus, $\tau_6$ is mapped on core 4 together with $\tau_7$. Since $\tau_6$ has the largest minimum inter-arrival time, the WCRT of $\tau_6$ is equal to $WCRT_6 = WCET_6 + WCET_7 = 9$.

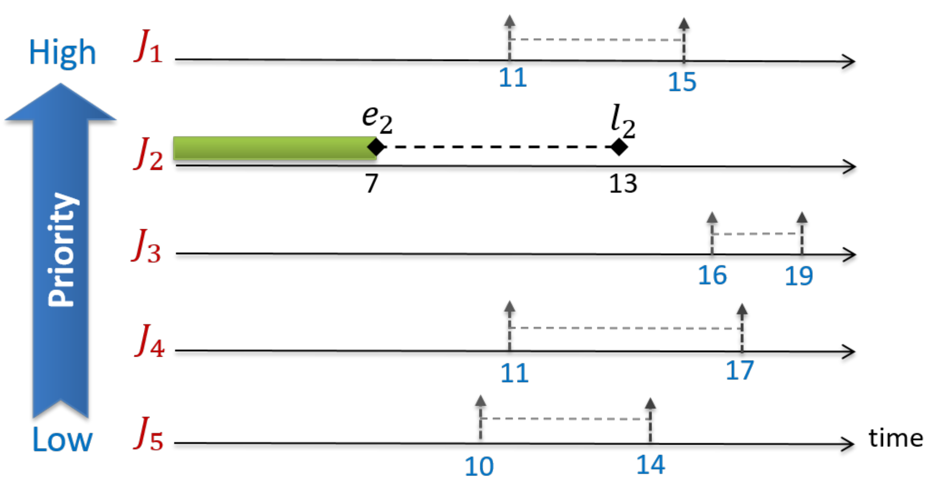
**Grading Rubric:**

- **2 points** for following a correct method (calculating and sorting utilizations).
- **3 points** if the partitioning is correct ($\tau_6$ and $\tau_7$ are together).
- **2 points** if the WCRT = 9 is correct.

## Question 5 [10 points]

Consider the following system with a set of non-preemptive jobs that are scheduled by a **work-conserving** *job-level fixed-priority scheduling policy.*

In the current state of the system, job $J_2$ has been dispatched and its earliest finish time $e_2 = 7$ and its latest finish time $l_2 = 13$. The following figure shows the earliest and latest release times of jobs $J = \{J_1, J_3, J_4, J_5\}$ that have not yet been dispatched by the scheduler. The upward arrows connected by the dashed gray lines show the potential release interval of each of these jobs.




(a) $\boxed{\text{3 points}}$ **Q5-1.** What is the latest possible time at which job $J_5$ can be dispatched "before any other job in $J$"? (namely, determine the latest start time = LST of $J_5$). Justify your answer.

**Solution:** $LST_5 = 14$ because it is the time at which the processor is certainly available (after finishing $J_2$) and a job is certainly released (job $J_5$).

Job $J_5$ will be a direct successor of $J_2$ when all other jobs are released as late as possible (i.e., $r_1 = 15$, $r_3 = 19$, $r_4 = 17$, and $r_5 = 14$).

(b) 4 points **Q5-2.** What is the latest start time of $J_4$ in a scenario in which $J_4$ is dispatched after $J_2$ and before any other job?
What is the release scenario that allows $J_4$ to be dispatched at the time you claim?

**Solution:** The answer is 14 because the scheduling policy is work-conserving, hence, at time 14 when the processor becomes available, it will certainly schedule a job (since job $J_5$ is certainly released at time 14. Hence, if job $J_4$ is going to be a direct successor of $J_2$ (namely, be dispatched before $J_5$), it must be dispatched before or at time 14. If both jobs $J_4$ and $J_5$ are released at time 14, $J_4$ will be dispatched since it has a higher priority than $J_5$.

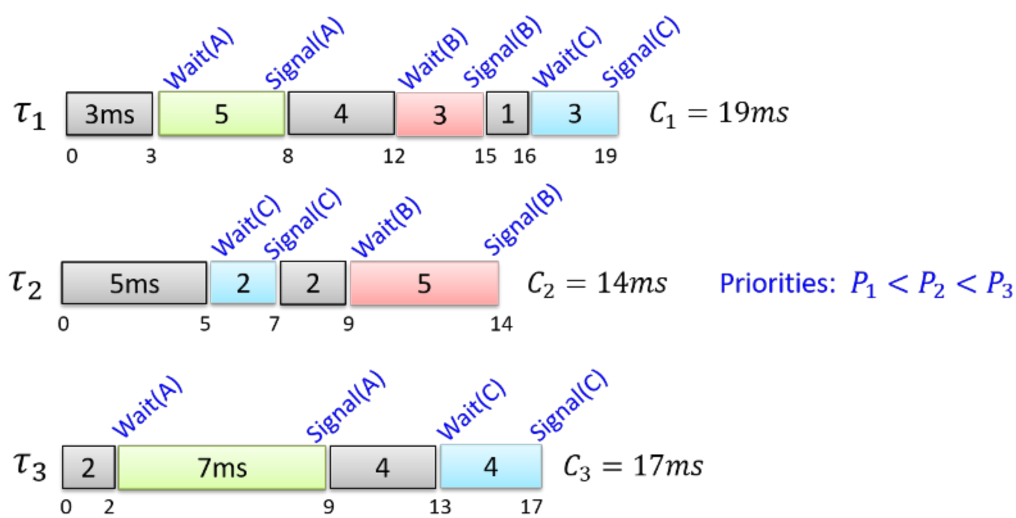
The following release pattern results in the dispatching of $J_4$ at time 14: $J_2$ finishes at 13, $r_1 = 15$, $r_3 = 19$, $r_4 = 14$, and $r_5 = 14$.

(c) 3 points **Q5-3.** Which job(s) do not have any chance to be dispatched on the platform directly after $J_2$? Why?

**Solution:** The answer is $J_3$ because its earliest release time is 16, however, by time 16, there are two jobs that have been certainly released in the system and are waiting for the processor to become available ($J_1$ and $J_5$). Since the scheduling policy is work-conserving, the scheduler will certainly dispatch a job on the platform at the latest by time 14. Hence, $J_3$ can never become a direct successor of $J_2$.

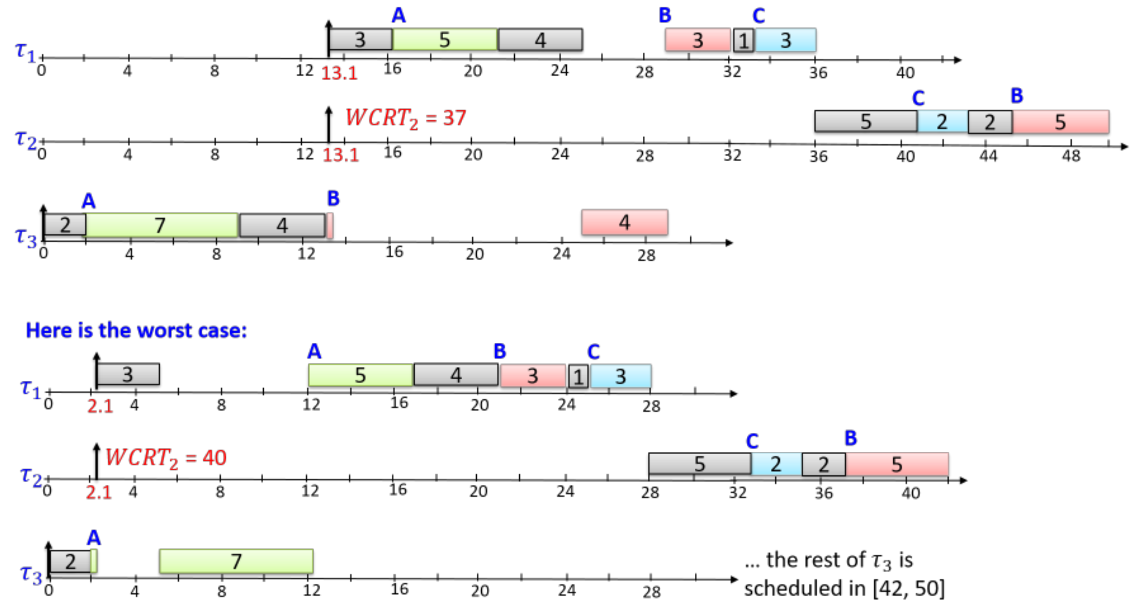
## Question 6 [15 points]

For the next three sub-questions, consider the following tasks. These tasks access shared resources that are protected by semaphores A, B, and C and are scheduled by fixed-priority scheduling on a uniprocessor platform. Assume that each of these tasks will release only 1 job. Note: the values in rectangles show the execution time of each segment of the task between two actions (wait or signal).



(a) 5 points **Q6-1.** What is the worst-case response time of $\tau_2$ if the shared resources are governed by the Higher-Locker Protocol (HLP)?
What are the release times of $\tau_1$, $\tau_2$, and $\tau_3$ that result in the worst-case response time for $\tau_2$?

**Solution:** WCRT: 40, $r_1 = 2$, $r_2 = 2$, and $r_3 = 0$.
The solution has been shown in the figure below.




**Grading Rubric: 2 points** for correct WCRT, and 3 points for a correct set of release times (for all tasks).

(b) 5 points **Q6-2.** Under Priority-Inheritance Protocol (PIP), which task(s) and on which semaphore(s) can directly or indirectly block task $\tau_1$? For each blocking, determine whether it is a direct blocking or indirect blocking.

**Solution:**

(a) $\tau_3$ can *directly* block $\tau_1$ on resource A

(b) $\tau_3$ can *directly* block $\tau_1$ on resource C

(c) $\tau_2$ can *directly* block $\tau_1$ on resource C

(d) $\tau_2$ can *directly* block $\tau_1$ on resource B

**Grading Rubric: -1 point** for each missing case.

(c) 5 points **Q6-3.** Find a case of "**indirect blocking**" for this task set under the **Priority-Inheritance Protocol (PIP)**. Explain which task indirectly blocks which other task on which resource. Provide the release times of $\tau_1$, $\tau_2$, and $\tau_3$ for that scenario.

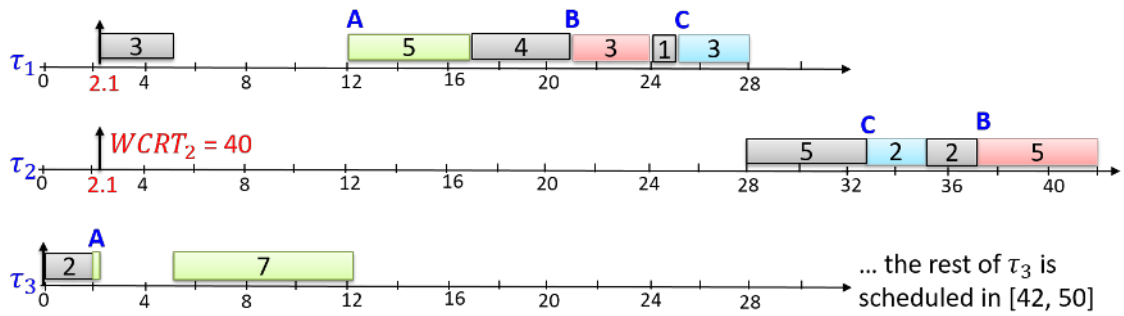
**Solution:**

(a) $\tau_3$ will indirectly block $\tau_2$ on resource A.

(b) $r_1 = 2 + \epsilon$.

(c) $r_2 = 2 + \epsilon$.

(d) $r_3 = 0$.




**Grading Rubric:** **2 points** for a correct identification of indirect blocking. **3 points** for a correct identification of release times.

## Question 7 [5 points]

Answer the next sub-questions for the following task code.

```
While(true)
{
    // ----- block 1 -------
    int temp = readTemperature();
    if (temp > 42)
        send(-1);
    else
    {
        int * array = read10Data();
        int max = -1;
        for (int i=0; i < 10; i++)
            if (max < 0 || array[i] > max)
                max = array[i];
        send(max);
    }
    // execution time of block1 is between 10 and 20
    sleep (50, ms);

    magicalFunction2();   // execution time is between 5 and 30
    sleep (100, ms);
}
```

(a) 3 points **Q7-1.** Briefly explain why this task is not periodic.

**Solution:** The task is sporadic because it's next activation depends on the time at which the last "sleep command" is executed which depends on how the task have been scheduled rather than on a constant time-triggered activation.

(b) 2 points **Q7-2.** What is the minimum-inter-arrival time between two jobs of this task? (note: assume that a job is released as soon as the last sleep function is completed).

**Solution:** It is $165 = 10$ (for block1) $+ 50$ (for the first sleep) $+ 5$ (for magicalFunction2) $+ 100$ (for the second sleep).