EXERCISE 2 REPORT

ECEN 5623

SUMMER 2021

UNIVERSITY OF COLORADO BOULDER

WRITTEN BY: SALONI SHAH

1. If you are using embedded Linux, make yourself an account on your R-Pi3b.

a**. Account creation** - To do this use “sudo adduser”, enter the well-known password, and enter user information as you see fit. Add your new user account as a “sudoer” using “visudo” right below root with the same privileges (if you need help with “vi”, here is a quick reference or reference card– use arrows to position cursor, below root hit Esc, “i” for insert, type username and privileges as above, and when done, Esc, “:”, “wq”). The old unix vi editor was one of the first full-screen visual editors – it still has the advantage of being found on virtually any Unix system in existence but is otherwise cryptic – along with Emacs it is still widely used in IT, by developers and systems engineers, so it is good to know the basics. If you really do not like vi or Emacs, your next best bet is “nano”, “VS code” or “geany” for Unix systems (you can normally do “sudo apt-get install geany” or any alternative editor you like best). You are welcome to use whatever editor works best for you in this class. Do a quick “sudo whoami” to demonstrate success for account creation.

Terminal commands:

sudo adduser <username>

- To add a new user on the system
- This command prompts the user to enter desired user information like name, phone number and address.

sudo visudo

- This command opens the file /etc/sudoer in vi editor
- Give ALL permissions to the desired user, under user privileges to grant root access to the user

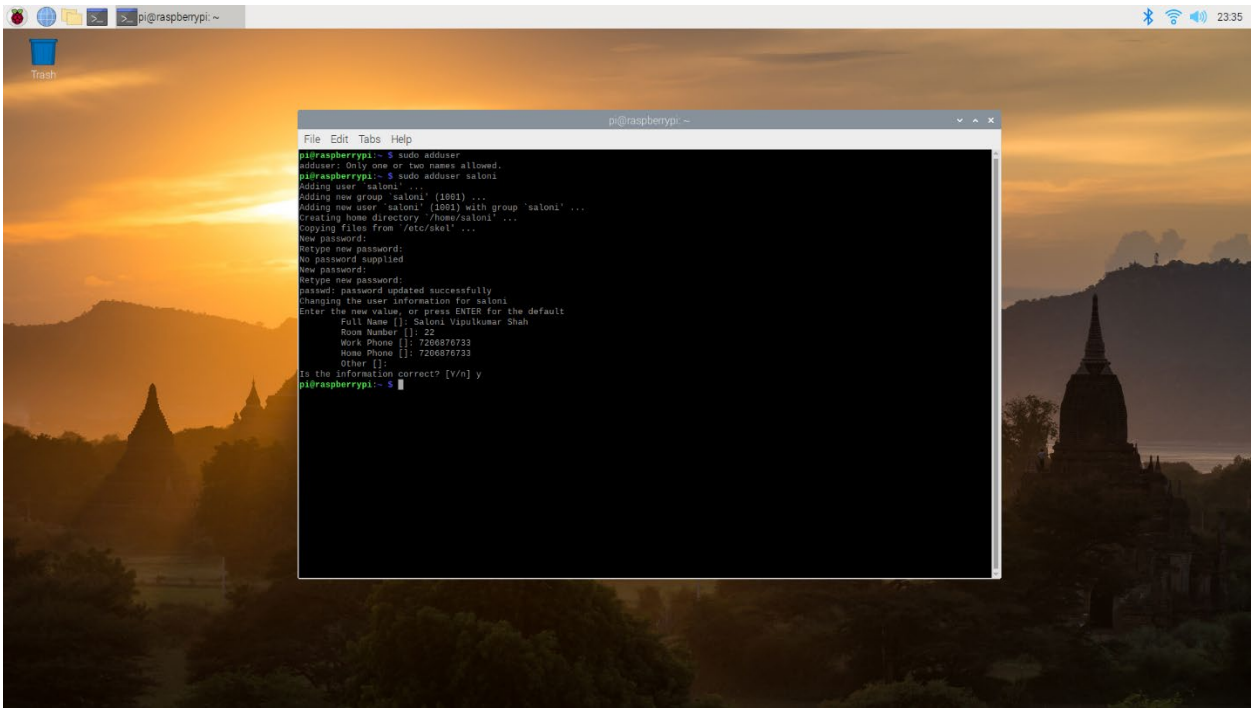
sudo whoami

- Shows the current user of the system or the user running the current command

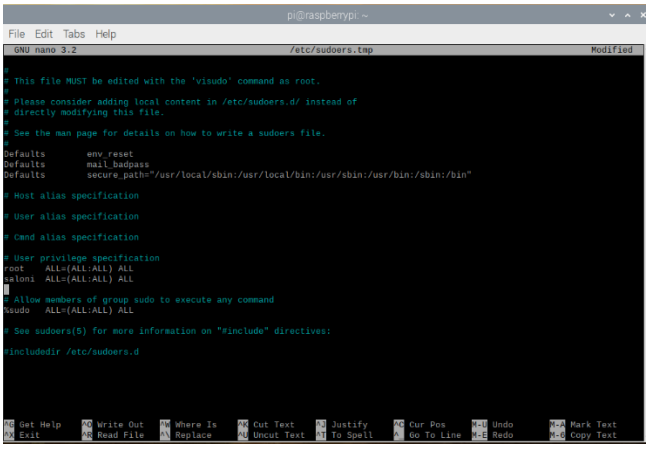
sudo apt-get install vim

- To install a service or application on the system. Here I have installed vim editor to write, edit, compile, and execute my code in the terminal itself.

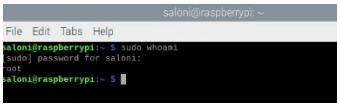
The below screenshot shows the desktop of my newly installed Raspbian OS and creation of a new user ‘saloni’ to the system.



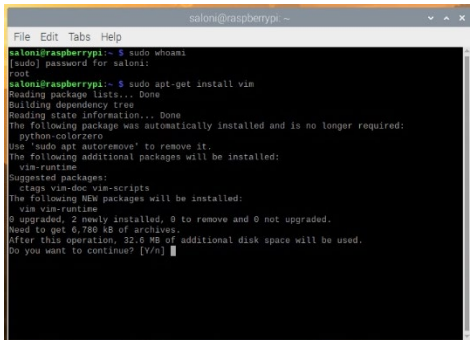
The below screenshot demonstrates granting full root access to the new user ‘saloni’ using sudoer.



The below screenshot demonstrates the use of whoami command to check user access.

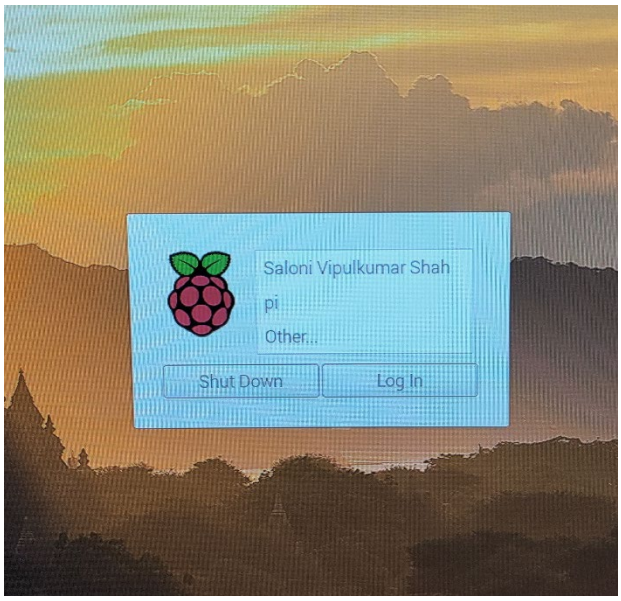


The below screenshot demonstrates the installation of vim editor using terminal.



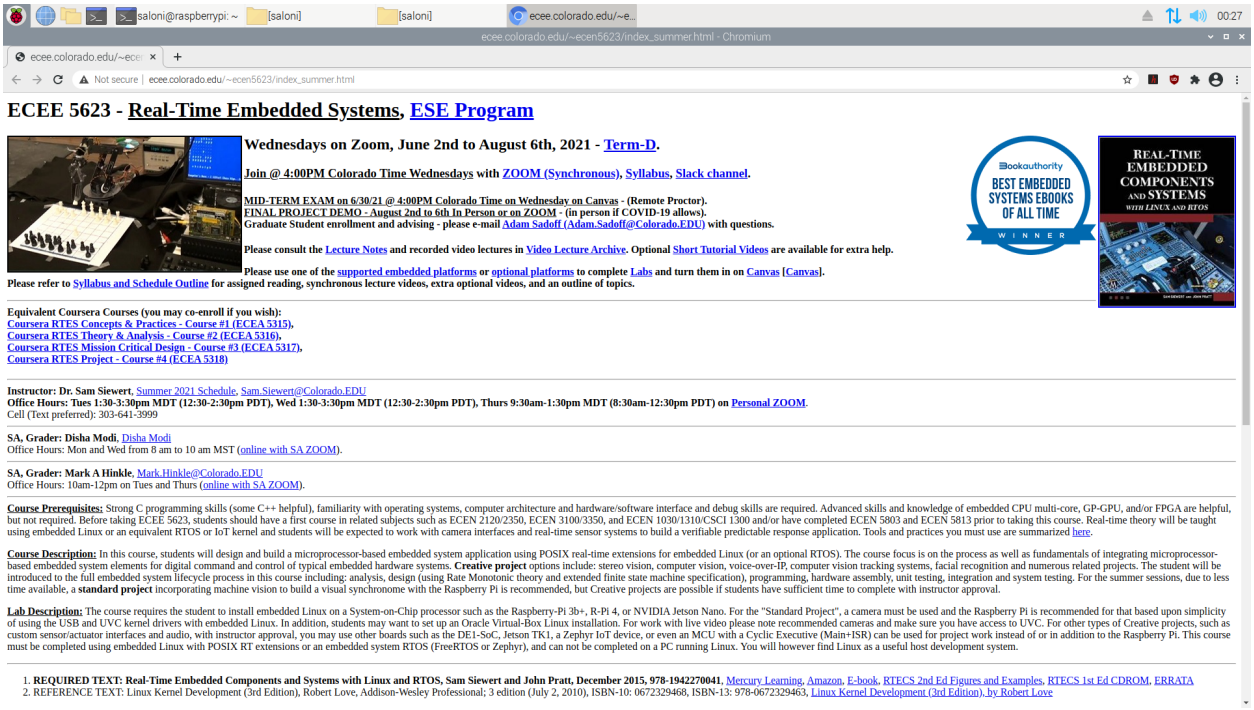
b. Logout and test your login, then logout. Use Alt+Print-Screen to capture your desktop and save as proof you set up your account. Note that you can always get a terminal with Ctrl+Alt+t key combination. Show evidence that you have done this with screenshots or photos with your smart phone.

The below screenshot shows login options for different users. After selection of a user, the system requires user password to access the system.

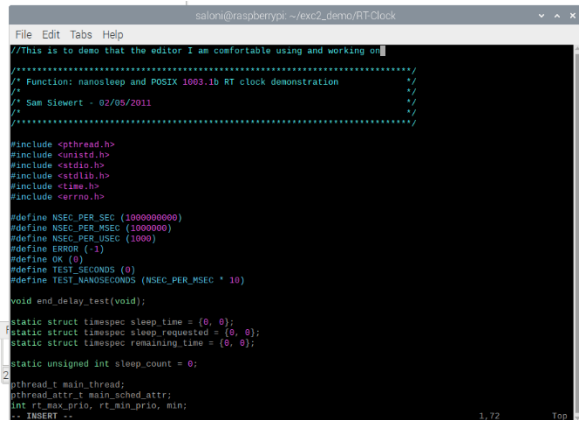


c. Make sure you can access our class web page on Firefox (or default browser) and set your home page to http://ecee.colorado.edu/~ecen5623/index_summer.html. Overall, make sure you are comfortable with development, debug, compiler general native or cross-development tools and document and demonstrate that you know them.

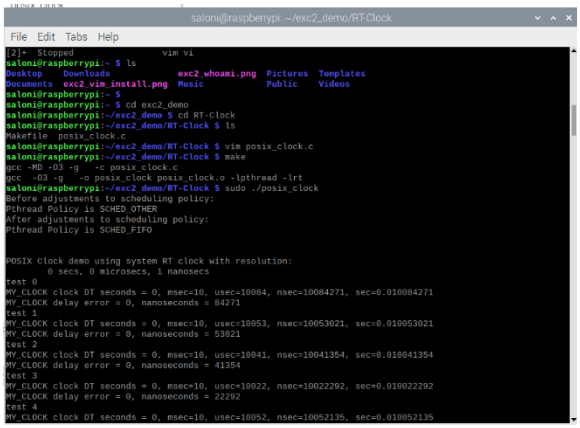
The below screenshot shows setting up ECEN 5623 RTES website as home page for the default browser. On start-up, the browser opens this webpage by default.



Here, I am using vim editor to successfully develop, debug, compile, and execute my code. Below screenshot shows the vim editor in terminal. Unlike other development IDEs, vim or other such editors have different functions or features to modify and save content like press 'I' for insert, ': wq' or ':x' to save and quit editor. Different such features to facilitate code modification.



Below I have compiled and executed an example code, to demonstrate my understanding of the working of the editor and terminal command lines.



2. Read the paper "Architecture of the Space Shuttle Primary Avionics Software System" [available on Canvas], by Gene Carlow.

a. Provide an explanation and critique of the frequency executive architecture.

- The system software of PASS has a carefully crafted design approach for synchronous response and to time the specific occurrences of the tasks in a cyclic loop.
- The implementation of such a system must be done very carefully and critically to synchronize the start and finish of different tasks.

- A type of cyclic executive, frequency executive architecture has been implemented in Guidance, Navigation and Control (GN&C) unit of avionic systems. This executive ensures that all flight control critical tasks are completed within 40 milliseconds of time limit.
- In GN&C, this executive completes the initialization and phasing of principal and associative functions.
- GN&C has 200 such principal functions included in 6 operational sequences (OPS). The execution rate of these principal functions varies from 25 Hz for flight control to 0.25 Hz for display unit.
- The cyclic executive is initiated by the OPS control segment. It is a table-driven dispatch design technique to process the principal functions, control and alter required executions.
- GN&C has three such executives based on the task frequency: high frequency executive, mid frequency executive and low frequency executive. High frequency executives are scheduled at high priority at 25 Hz rate to process principal functions related to flight control. Mid frequency and low frequency executive are scheduled at a lower priority for tasks with a rate of 6.25 Hz to 0.25 Hz.

b. What advantages and disadvantages does the frequency executive have compared to the real-time threading and tasking implementation methods for real-time software systems? Please be specific about the advantages and disadvantages and provide at least 3 advantages as well as 3 disadvantages.

Advantages of Frequency Executive Architecture:

- It is a synchronous system. The dispatch of each application is perfectly timed beforehand to develop a synchronous system. All the applications occur at a specific point of time after the start of the system. Unlike threading and tasking implementation where the task priority is dynamic, in frequency cyclic executive, all tasks are run at a particular time. It prevents task overruns and makes the system predictable.
- It is a deterministic system. If one cyclic execution of the applications is done successfully without missing any deadlines, then most likely the entire

system execution will be done successfully. The overall working of all the applications can be predicted by examining one cycle.

- It is a simple process. The hierarchical execution of tasks is a simple cycle or loop which can be debugged very easily.
- It has less context switching time. All applications in GN&C do not support preemption. This reduces the context switching time between different tasks. Whereas in threading or tasking, preemption of lower priority task by a high priority task involves some context switching.

Disadvantages of Frequency Executive Architecture:

- It has lot of CPU utilization. Continuously running the applications in a cycle or loop increases the utilization of resources. This reduces margin for system safety.
- It is an inflexible system. Since all applications are running at specific time pre-determined from the start of the system, it would be very difficult to add any other task to the cycle. Hence the software has very limited chances of implementing any changes. To add any critical task to the system, the cyclic executive need to be programmed again.
- It has fixed priority scheduling. Since the task priority depends on its frequency, the system will not be able to accommodate a critical but low frequency task. If there is a low frequency critical task related to flight control or other critical system, it will not preempt other higher frequency but lower priority task.

3. Download feasibility example code and build it on your Raspberry Pi or alternate system of your choice and execute the code.

   The feasibility code checks for RM policy feasibility for different service sets using RM LUB condition, Completion feasibility test and Scheduling Point feasibility test.

RM LUB Feasibility Condition

According to Liu and Layland, an RM policy is feasible for a service set if the total CPU utilization by all the services is less than or equal to RM Least Upper Bound where

$$\text{RM LUB} = m(2^{\frac{1}{m}} - 1)$$

Where m = number of services

This is an easy but pessimistic approach to check system feasibility as it leaves a lot of CPU margin which is not required in many applications. This condition can reject a policy which might be completely feasible. Hence this is a sufficient test for policy feasibility but not necessary. To overcome the shortcomings of this approach, two different, necessary and sufficient (N&S) feasibility tests were introduced which are much more accurate.

Scheduling Point test

According to Lehoczky, Sha and Ding, if a set of services can be scheduled without missing any deadlines from a critical point to the longest deadline time of all the tasks, then the schedule is feasible. Based on this theorem they developed a feasibility test – Scheduling Point Test. This test is based on the following formulas:

$$\forall i, 1 \le i \le n, \min \sum_{j=1}^{i} C_j \left\lceil \frac{(l)T_k}{T_j} \right\rceil \le (l)T_k$$

$$(k,l) \in R_i$$

$$R_i = \left\{ (k,l) \middle| 1 \le k \le i, l = 1, \ldots, \left\lfloor \frac{T_i}{T_k} \right\rfloor \right\}$$

- Where n is the number of tasks in the set S_i to S_n, where S_1 has higher priority than S_2, and S_n has higher priority than $S_{n>1}$.
- j identifies S_j, a service in the set between S_1 and S_n.
- k identifies S_k, a service whose l periods must be analyzed.
- l represents the number of periods of S_k to be analyzed.
- $\left\lceil \frac{(l)T_k}{T_j} \right\rceil$ represents the number of times S_j executes within l period of S_k.

If the above-mentioned conditions are satisfied, then the RM policy for that service set is feasible

Completion Feasibility Test

An alternative of the Scheduling Point test is completion test which is based on the following formulas:

$$a_n(t) = \sum_{j=1}^{n} \left\lceil \frac{t}{T_j} \right\rceil C_j$$

- $\left\lceil \frac{t}{T_j} \right\rceil$ is the number of executions of S_j at time t.
- $\left\lceil \frac{t}{T_j} \right\rceil C_j$ is the demand of S_j in time at t.
- $a_n(t)$ is the total cumulative demand from the n tasks up to time t.

If the value of a_n(t) is less than or equal to the deadline for S_n, then the service is feasible. Similarly, proving this for all the services will make the service set feasible.

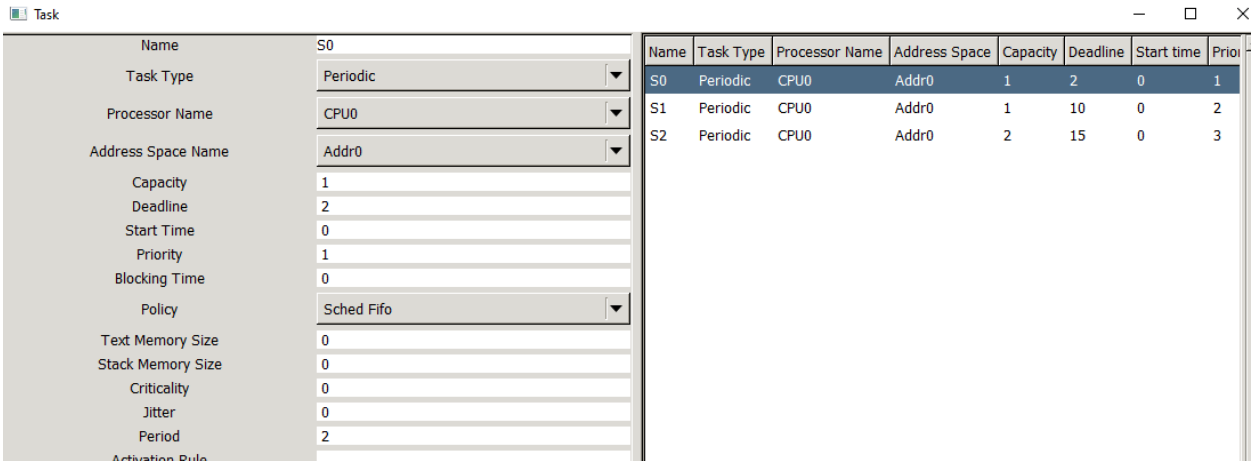
Cheddar Tool

For further analysis of the given examples and implement different scheduling policies, the Cheddar tool was used. Below is a small synopsis of the working of the tool

Cheddar is a tool to study the behavior of different real time system applications. This tool was developed at University of Brest in France. It is a free, available for all real time system simulator and feasibility testing tool. In Cheddar, a service is defined by a set of core, processor, buffer, address, shared resource, and tasks. A task is defined by its period, deadline, and capacity. For any service set, a scheduling simulation and feasibility test can be done. Cheddar also

provides most schedulers like EDF and LLF and queueing policies like SCHED_FIFO, SCHED_OTHER etc. Cheddar also supports many other features related to memory usage, buffers, cache, and other analysis which is not in the scope of this course.

a. Compare the tests provided to analysis using Cheddar for the first 4 examples tested by the code.



EXAMPLE 0:

Example 0	Service	Period		Freq f	f0 multiple	WCET		Utility							
	S1	T1	2	0.5	7.5	C1	1	U1	50.00%	LCM =	30				
	S2	T2	10	0.1	1.5	C2	1	U2	10.00%	LUB =	77.98%				
	S3	T3	15	0.066667	1	C3	2	U3	13.33%	Utot =	73.33%	Slack	26.67%	Slack+U	100.00%

RATE MONOTONIC SCHEDULING POLICY TIMING DIAGRAM

By observing the rate monotonic timing diagram, it can be concluded that the RM policy is schedulable without missing any deadlines

CHEDDAR TOOL RATE MONOTONIC POLICY OUTPUT

Scheduling simulation, Processor CPU0 :
- Number of context switches : 14
- Number of preemptions : 2

- Task response time computed from simulation :
  S0 => 1/worst
  S1 => 2/worst
  S2 => 6/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.

Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 30.0, see Leung and Merill (1980) from [21].
- 8 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor : 1.
- Processor utilization factor with deadline is 0.73333 (see [1], page 6).
- Processor utilization factor with period is 0.73333 (see [1], page 6).
- In the preemptive case, with RM, the task set is schedulable because the processor utilization factor 0.73333 is equal or less than 0.77976 (see [1], page 16, theorem 8).

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time : (see [2], page 3, equation 4).
  S0 => 1
  S1 => 2
  S2 => 6
- All task deadlines will be met : the task set is schedulable.

As expected from the tool output, the RM policy is schedulable which is the conclusion derived from the timing diagram. The cheddar tool feasibility test is done based on processor utilization as well as worst case timing response. The RM LUB condition is also satisfied, and the schedule is feasible based on the timing analysis.

FEASIBILITY CODE – COMPLETION TEST OUTPUT

```
Ex-0 U=73.33% (C1=1, C2=1, C3=2; T1=2, T2=10, T3=15; T=D):
CT test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=10.000000, utility_sum = 0.600000
for 2, wcet=2.000000, period=15.000000, utility_sum = 0.733333
utility_sum = 0.733333
LUB = 0.779763
RM LUB FEASIBLE
```

FEASIBILITY CODE – SCHEDULING POINT TEST OUTPUT

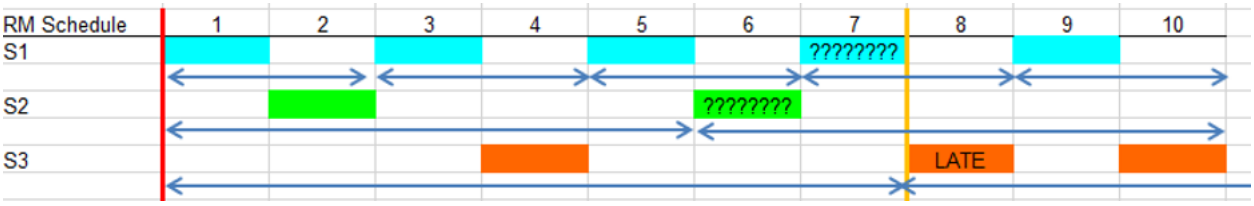
```
Ex-0 U=73.33% (C1=1, C2=1, C3=2; T1=2, T2=10, T3=15; T=D):
SP test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=10.000000, utility_sum = 0.600000
for 2, wcet=2.000000, period=15.000000, utility_sum = 0.733333
utility_sum = 0.733333
LUB = 0.779763
RM LUB FEASIBLE
```

The feasibility code computes the RM LUB and carries out necessary and sufficient feasibility tests – completion test and scheduling point test to demonstrate whether the rate monotonic policy is schedulable for that particular set of services. As per the code, the RM policy is feasible as it passed the feasibility test as well as RM LUB condition devised by Liu and Layland. This output aligns with the output from timing analysis and cheddar tool.

EXAMPLE 1:

Example 1	Service	Period		Freq f	f0 multiple	WCET		Utility			
	S1	T1	2	0.5	3.5	C1	1	U1	50.00%	**LCM =**	**70**
	S2	T2	5	0.2	1.4	C2	1	U2	20.00%	**LUB =**	**77.98%**
	S3	T3	7	0.142857	1	C3	2	U3	28.57%	**Utot =**	**98.57%**

RATE MONOTONIC SCHEDULING POLICY TIMING DIAGRAM



For the given set of services, according to the timing diagram analysis, Rate Monotonic policy is not feasible as it misses a few deadlines.

CHEDDAR TOOL RATE MONOTONIC POLICY OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  68
- Number of preemptions :  10

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 2/worst
   S2 => 8/worst ,  missed its deadline (absolute deadline =  7 ; completion time =  8)
- Some task deadlines will be missed : the task set is not schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 70.0, see Leung and Merill (1980) from [21].
- 1 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 0.98571 (see [1], page 6).
- Processor utilization factor with period is 0.98571 (see [1], page 6).
- In the preemptive case, with RM, we cannot prove that the task set is schedulable because the processor utilization factor 0.98571 is more than 0.77976 (


2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :  (see [2], page 3, equation 4).
   S0 => 1
   S1 => 2
   S2 => 8,  missed its deadline (deadline =  7)
- Some task deadlines will be missed : the task set is not schedulable.
```

The cheddar tool output is correct as expected. It gives an analysis of when the service deadline will be missed. Based on this the feasibility test is carried out. For this service set, the RM policy is not feasible.

FEASIBILITY CODE – COMPLETION TEST OUTPUT

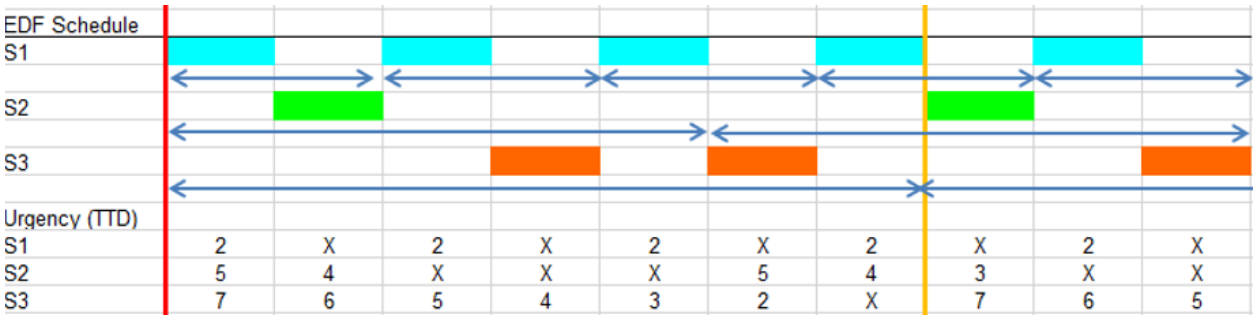
```
Ex-1 U=98.57% (C1=1, C2=1, C3=2; T1=2, T2=5, T3=7; T=D):
CT test INFEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=5.000000, utility_sum = 0.700000
for 2, wcet=2.000000, period=7.000000, utility_sum = 0.985714
utility_sum = 0.985714
LUB = 0.779763
RM LUB INFEASIBLE
```

FEASIBILITY CODE – SCHEDULING POINT TEST OUTPUT

```
Ex-1 U=98.57% (C1=1, C2=1, C3=2; T1=2, T2=5, T3=7; T=D):
SP test INFEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=5.000000, utility_sum = 0.700000
for 2, wcet=2.000000, period=7.000000, utility_sum = 0.985714
utility_sum = 0.985714
LUB = 0.779763
RM LUB INFEASIBLE
```

Similar to the output from timing analysis and cheddar tool, the feasibility code carries out the tests and checks RM LUB condition. As proven by the other tools, the RM policy is not feasible. Hence, further we check whether Earliest Deadline First (EDF) and Least Laxity First (LLF) scheduling policies are feasible for the given services.

EARLIEST DEADLINE FIRST TIMING DIAGRAM



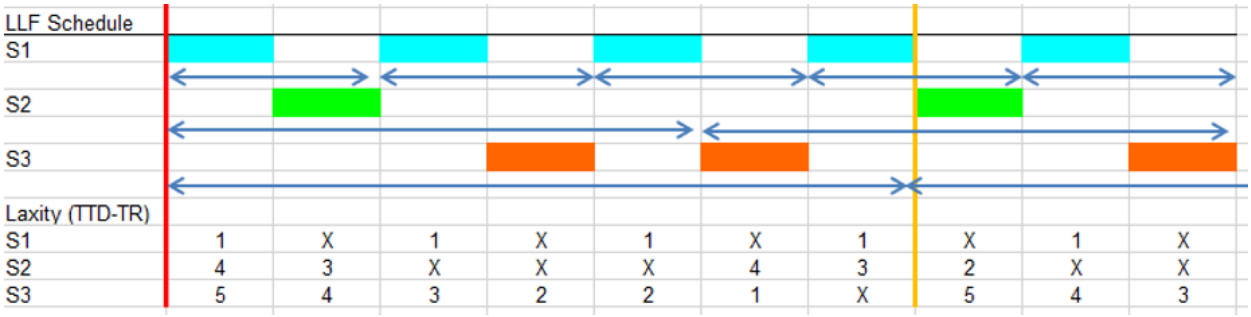
According to the timing diagram, using dynamic priority scheduling policy, the deadlines are not missed, and the schedule is feasible.

CHEDDAR TOOL EARLIEST DEADLINE FIRST OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  68
- Number of preemptions :  10

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 4/worst
   S2 => 6/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 70.0, see Goossens and Devillers (1997) from [21].
- 1 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 0.98571 (see [1], page 6).
- Processor utilization factor with period is 0.98571 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 0.98571 is equal or less than 1.00000 (

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :
   S0 => 1
   S1 => 4
   S2 => 6
- All task deadlines will be met : the task set is schedulable.
```

Similar to timing analysis, the cheddar tool tests the service test feasibility using processor utilization factor and the worst-case response time analysis. The simulation and feasibility test proves that EDF policy is feasible for the given services.

LEAST LAXITY FIRST TIMING DIAGRAM

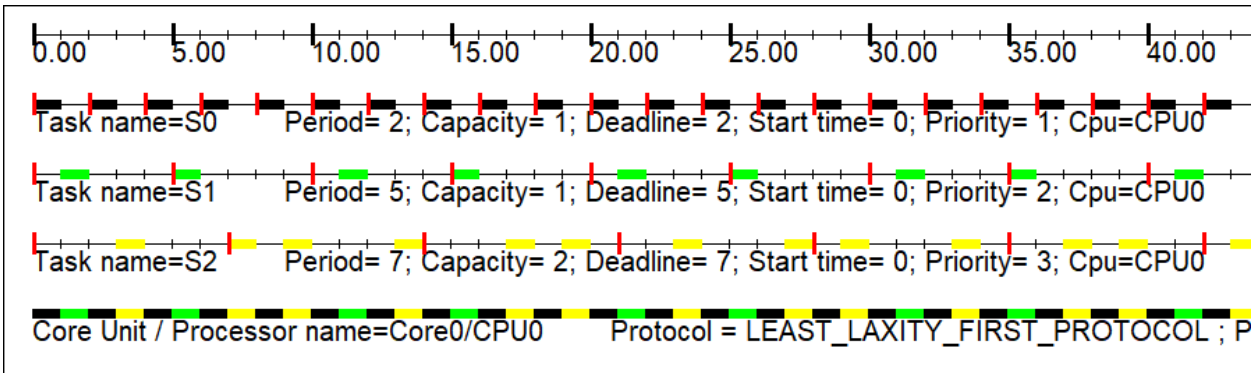


By observing the timing diagram for LLF, we can conclude that the LLF policy is feasible.

CHEDDAR TOOL – LEAST LAXITY FIRST OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  68
- Number of preemptions :  10

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 2/worst
   S2 => 8/worst ,  missed its deadline (absolute deadline =  7 ; completion time =  8)
- Some task deadlines will be missed : the task set is not schedulable.



Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 70.0, see Leung and Merrill (1980) from [21].
- 1 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 0.98571 (see [1], page 6).
- Processor utilization factor with period is 0.98571 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 0.98571 is equal or less than 1.00000

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :
   S0 => 1
   S1 => 4
   S2 => 6
- All task deadlines will be met : the task set is schedulable.
```

The cheddar tool analysis gives the same output as timing diagram i.e. the policy is feasible. However there is a glitch in the tool output analysis. The scheduling policy simulation result indicates that a few deadlines are missed which is clearly not the case.

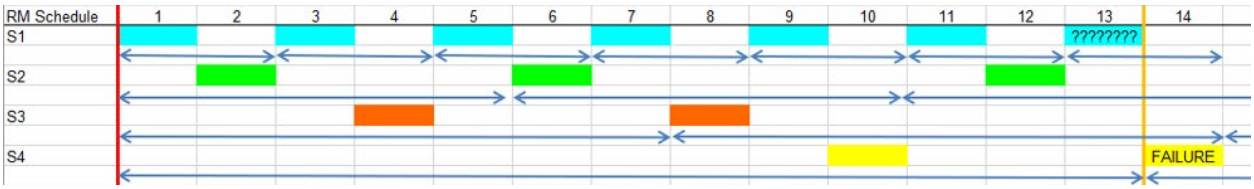


As we can observe in the above screenshot of timing analysis done by cheddar tool for LLF policy, there are a few mistakes in the diagram. Instead of LLF policy, the diagram resembles the timing diagram of RM policy. Hence, the timing diagram analysis is not accurate for LLF in cheddar.

EXAMPLE 2:

Example 2	Service	Period		Freq f	f0 multiple	WCET		Utility			
	S1	T1	2	0.5	6.5	C1	1	U1	50.00%	**LCM =**	**910**
	S2	T2	5	0.2	2.6	C2	1	U2	20.00%		
	S3	T3	7	0.142857	1.857143	C3	1	U3	14.29%	**LUB =**	**75.68%**
	S4	T4	13	0.076923	1	C4	2	U4	15.38%	**Utot =**	**99.67%**

RATE MONOTONIC SCHEDULING POLICY TIMING DIAGRAM



For the given set of services, according to the timing diagram analysis, Rate Monotonic policy is not feasible as it misses a few deadlines.

CHEDDAR TOOL – RATE MONOTONIC POLICY OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  904
- Number of preemptions :  70

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 2/worst
   S2 => 4/worst
   S4 => 16/worst ,  missed its deadline (absolute deadline =  13 ; completion time =  14),  missed its deadline (absolute deadline =  26 ; completion time =  28),  missed its deadline (absolute deadline =  39 ; completion time =  40),
- Some task deadlines will be missed : the task set is not schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 910.0, see Leung and Merill (1980) from [21].
- 3 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 0.99670 (see [1], page 6).
- Processor utilization factor with period is 0.99670 (see [1], page 6).
- In the preemptive case, with RM, we cannot prove that the task set is schedulable because the processor utilization factor 0.99670 is more than 0.75683 (see [1], page 16, theorem 8).

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :  (see [2], page 3, equation 4).
   S0 => 1
   S1 => 2
   S2 => 4
   S4 => 16,  missed its deadline (deadline =  13)
- Some task deadlines will be missed : the task set is not schedulable.
```

The cheddar tool output is correct as expected. It gives an analysis of when the service deadline will be missed. Based on this the feasibility test is carried out. For this service set, the RM policy is not feasible.

FEASIBILITY CODE – COMPLETION TEST OUTPUT

```
Ex-2 U=99.67% (C1=1, C2=1, C3=1, C4=2; T1=2, T2=5, T3=7, T4=13; T=D):
CT test INFEASIBLE
for 4, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=5.000000, utility_sum = 0.700000
for 2, wcet=1.000000, period=7.000000, utility_sum = 0.842857
for 3, wcet=2.000000, period=13.000000, utility_sum = 0.996703
utility_sum = 0.996703
LUB = 0.756828
RM LUB INFEASIBLE
```

FEASIBILITY CODE – SCHEDULING POINT TEST OUTPUT

```
Ex-2 U=99.67% (C1=1, C2=1, C3=1, C4=2; T1=2, T2=5, T3=7, T4=13; T=D):
SP test INFEASIBLE
for 4, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=5.000000, utility_sum = 0.700000
for 2, wcet=1.000000, period=7.000000, utility_sum = 0.842857
for 3, wcet=2.000000, period=13.000000, utility_sum = 0.996703
utility_sum = 0.996703
LUB = 0.756828
RM LUB INFEASIBLE
```

Similar to the output from timing analysis and cheddar tool, the feasibility code carries out the tests and checks RM LUB condition. As proven by the other tools, the RM policy is not feasible.

EARLIEST DEADLINE FIRST TIMING DIAGRAM

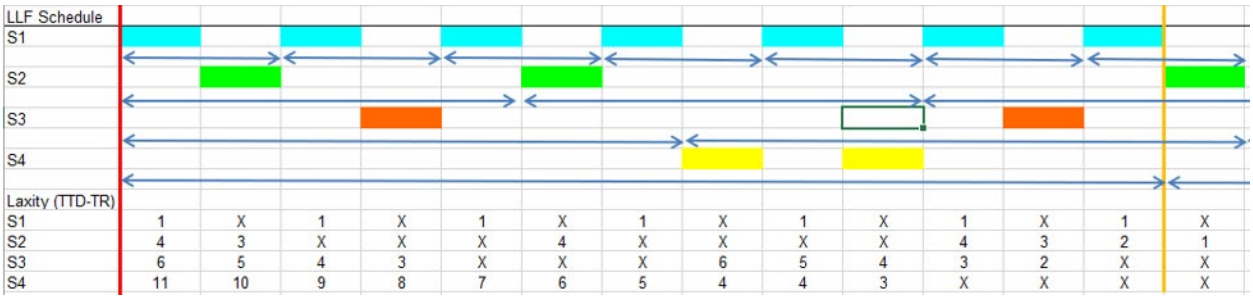
According to the timing diagram, using dynamic priority scheduling policy, the deadlines are not missed and the schedule is feasible.

CHEDDAR TOOL EARLIEST DEADLINE FIRST OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  904
- Number of preemptions :  70

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 4/worst
   S2 => 6/worst
   S4 => 12/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 910.0, see Goossens and Devillers (1997) from [21].
- 3 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 0.99670 (see [1], page 6).
- Processor utilization factor with period is 0.99670 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 0.99670 is equal or less than 1.0000

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :
   S0 => 1
   S1 => 4
   S2 => 6
   S4 => 12
- All task deadlines will be met : the task set is schedulable.
```

Similar to timing analysis, the cheddar tool tests the service test feasibility using processor utilization factor and the worst-case response time analysis. The simulation and feasibility test proves that EDF policy is feasible for the given services.

LEAST LAXITY FIRST TIMING DIAGRAM

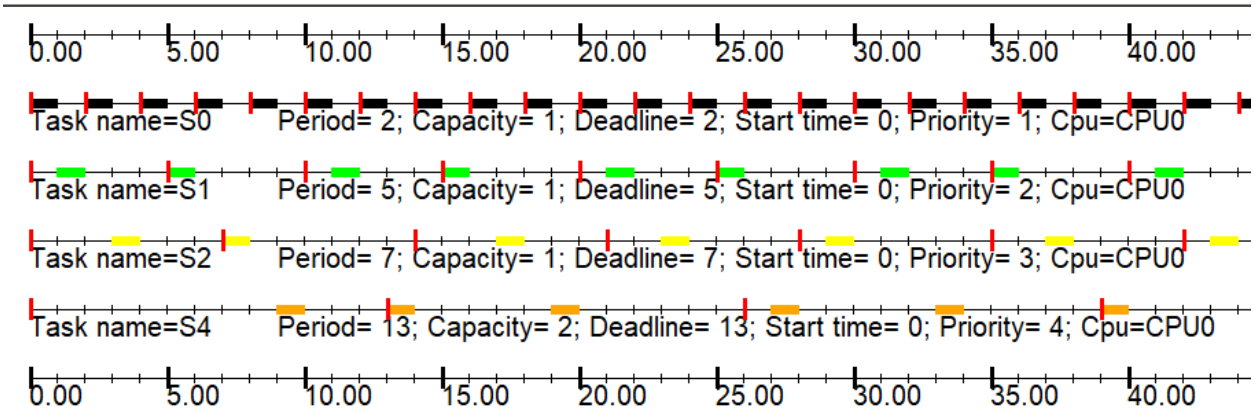


By observing the timing diagram for LLF, we can conclude that the LLF policy is feasible.

CHEDDAR TOOL LEAST LAXITY FIRST OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  904
- Number of preemptions :  70

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 2/worst
   S2 => 4/worst
   S4 => 16/worst ,  missed its deadline (absolute deadline =  13 ; completion time =  14),  missed its deadline (absolute deadline =  26 ; completion time =  28),  missed its deadline (absolute deadline =  39 ; completion time =  40),  misse
- Some task deadlines will be missed : the task set is not schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 910.0, see Leung and Merrill (1980) from [21].
- 3 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 0.99670 (see [1], page 6).
- Processor utilization factor with period is 0.99670 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 0.99670 is equal or less than 1.00000 (see [7]).

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :
   S0 => 1
   S1 => 4
   S2 => 6
   S4 => 12
- All task deadlines will be met : the task set is schedulable.
```

The cheddar tool analysis gives the same output as timing diagram i.e., the policy is feasible. However, there is a glitch in the tool output analysis. The scheduling policy simulation result indicates that a few deadlines are missed which is clearly not the case.

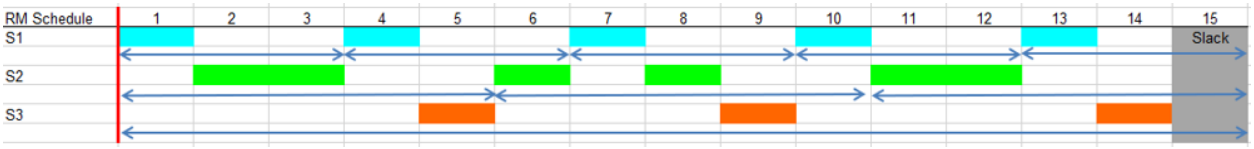


As we can observe in the above screenshot of timing analysis done by cheddar tool for LLF policy, there are a few mistakes in the diagram. Instead of LLf policy, the diagram resembles the timing diagram of RM policy. Hence, the timing diagram analysis is not accurate for LLF in cheddar.

EXAMPLE 3:

Example 3	Services	Freq f	f0 multiple	Period		WCET		Utility							
	S1	0.333333	5	T1	3	C1	1	U1	0.33	LCM =	15				
	S2	0.2	3	T2	5	C2	2	U2	0.4	LUB =	77.98%				
	S3	0.066667	1	T3	15	C3	3	U3	0.2	Utot =	93.33%	Slack	6.67%	Slack+U	100.00%

RATE MONOTONIC SCHEDULING POLICY TIMING DIAGRAM



By observing the rate monotonic timing diagram, it can be concluded that the RM policy is schedulable without missing any deadlines.

CHEDDAR TOOL RATE MONOTONIC POLICY OUTPUT

Scheduling simulation, Processor CPU0 :
- Number of context switches : 11
- Number of preemptions : 3

- Task response time computed from simulation :
  S0 => 1/worst
  S1 => 3/worst
  S2 => 14/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.

Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 15.0, see Leung and Merill (1980) from [21].
- 1 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor : 1.
- Processor utilization factor with deadline is 0.93333 (see [1], page 6).
- Processor utilization factor with period is 0.93333 (see [1], page 6).
- In the preemptive case, with RM, we cannot prove that the task set is schedulable because the processor utilization factor 0.93333 is more than 0.77976 (

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time : (see [2], page 3, equation 4).
  S0 => 1
  S1 => 3
  S2 => 14
- All task deadlines will be met : the task set is schedulable.

As expected from the tool output, the RM policy is schedulable which is the conclusion derived from the timing diagram. The cheddar tool feasibility test is done based on processor utilization as well as worst case timing response. Although the RM LUB condition is not satisfied, the schedule is feasible based on the timing analysis.

FEASIBILITY CODE – COMPLETION TEST OUTPUT

```
Ex-3 U=93.33% (C1=1, C2=2, C3=3; T1=3, T2=5, T3=15; T=D):
CT test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=3.000000, utility_sum = 0.333333
for 1, wcet=2.000000, period=5.000000, utility_sum = 0.733333
for 2, wcet=3.000000, period=15.000000, utility_sum = 0.933333
utility_sum = 0.933333
LUB = 0.779763
RM LUB INFEASIBLE
```

FEASIBILITY CODE – SCHEDULING POINT TEST OUTPUT

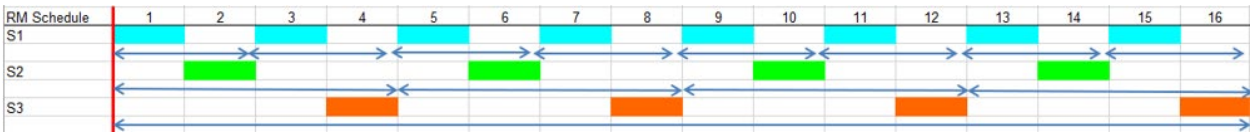
```
Ex-3 U=93.33% (C1=1, C2=2, C3=3; T1=3, T2=5, T3=15; T=D):
SP test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=3.000000, utility_sum = 0.333333
for 1, wcet=2.000000, period=5.000000, utility_sum = 0.733333
for 2, wcet=3.000000, period=15.000000, utility_sum = 0.933333
utility_sum = 0.933333
LUB = 0.779763
RM LUB INFEASIBLE
```

The feasibility code computes the RM LUB and also carries out necessary and sufficient feasibility tests – completion test and scheduling point test to demonstrate whether the rate monotonic policy is schedulable for that particular set of services. As per the code, the RM policy is feasible based on the feasibility tests. However, it does not satisfy the RM LUB condition and hence deems the scheduling policy infeasible. The RM LUB condition is a pessimistic feasibility test which can consider a perfectly feasible policy to be infeasible. Hence the code output is partially in agreement with our analysis.

EXAMPLE 4:

Example 4	Services	Freq f	f0 multiple	Period		WCET		Utility			
	S1	0.5	8	T1	2	C1	1	U1	0.5	LCM =	16
	S2	0.25	4	T2	4	C2	1	U2	0.25	LUB =	77.98%
	S3	0.0625	1	T3	16	C3	4	U3	0.25	Utot =	100.00%

RATE MONOTONIC SCHEDULING POLICY TIMING DIAGRAM



By observing the rate monotonic timing diagram, it can be concluded that the RM policy is schedulable without missing any deadlines.

CHEDDAR TOOL RATE MONOTONIC POLICY OUTPUT

Scheduling simulation, Processor CPU0 :
- Number of context switches : 15
- Number of preemptions : 3

- Task response time computed from simulation :
  S0 => 1/worst
  S1 => 2/worst
  S2 => 16/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.

Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 16.0, see Leung and Merill (1980) from [21].
- 0 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor : 1.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with RM, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time : (see [2], page 3, equation 4).
  S0 => 1
  S1 => 2
  S2 => 16
- All task deadlines will be met : the task set is schedulable.

As expected from the tool output, the RM policy is schedulable which is the conclusion derived from the timing diagram. The cheddar tool feasibility test is done based on processor utilization as well as worst case timing response. However, the feasibility test of cheddar is not accurate. The RM LUB computation done to check processor utilization is incorrect (actual RM LUB is 0.7797). Although the RM LUB condition is not satisfied, the feasibility test of cheddar gives misleading output. This analysis was done by hand evaluation of RM policy.

FEASIBILITY CODE – COMPLETION TEST OUTPUT

```
Ex-4 U=100.00% (C1=1, C2=1, C3=4; T1=2, T2=4, T3=16; T=D):
CT test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=4.000000, utility_sum = 0.750000
for 2, wcet=4.000000, period=16.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.779763
RM LUB INFEASIBLE
```

FEASIBILITY CODE – SCHEDULING POINT TEST OUTPUT

```
Ex-4 U=100.00% (C1=1, C2=1, C3=4; T1=2, T2=4, T3=16; T=D):
SP test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=4.000000, utility_sum = 0.750000
for 2, wcet=4.000000, period=16.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.779763
RM LUB INFEASIBLE
```

As per the code, the RM policy is feasible based on the feasibility tests. However, it does not satisfy the RM LUB condition and hence deems the scheduling policy infeasible. Hence the code output is partially in agreement with our analysis.

b. Now, implement remaining examples [6 more] of your interest from those that we reviewed in class (found here). Complete analysis using Cheddar RM. In cases where RM fails, test EDF or LLF to see if it succeeds and if it does, explain why. Cheddar uses both service simulations over the LCM of the periods as well as feasibility analysis based on the RM LUB and scheduling-point/completion-test algorithms, referred to as “Worst Case Analysis”.

To further test the accuracy of cheddar tool and feasibility code, additional 6 service set examples were implemented. I have carried out detailed analysis of their outputs below.

EXAMPLE 5:

Example 0	Service	Period		Freq f	f0 multiple	WCET		Utility							
	S1	T1	2	0.5	7.5	C1	1	U1	50.00%	**LCM =**	**30**				
	S2	T2	5	0.2	3	C2	1	U2	20.00%	**LUB =**	**77.98%**				
	S3	T3	15	0.066667	1	C3	2	U3	13.33%	**Utot =**	**83.33%**	**Slack**	**16.67%**	**Slack+U**	**100.00%**

RATE MONOTONIC SCHEDULING POLICY TIMING DIAGRAM

By observing the rate monotonic timing diagram, it can be concluded that the RM policy is schedulable without missing any deadlines.

CHEDDAR TOOL RATE MONOTONIC POLICY OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  20
- Number of preemptions :  2

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 2/worst
   S2 => 8/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 30.0, see Leung and Merill (1980) from [21].
- 5 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 0.83333 (see [1], page 6).
- Processor utilization factor with period is 0.83333 (see [1], page 6).
- In the preemptive case, with RM, we cannot prove that the task set is schedulable because the processor utilization factor 0.83333 is more than 0.77976


2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :  (see [2], page 3, equation 4).
   S0 => 1
   S1 => 2
   S2 => 8
- All task deadlines will be met : the task set is schedulable.
```

As expected from the tool output, the RM policy is schedulable which is the conclusion derived from the timing diagram. The cheddar tool feasibility test is done based on processor utilization as well as worst case timing response. Although the RM LUB condition is not satisfied, the schedule is feasible based on the timing analysis.

FEASIBILITY CODE – COMPLETION TEST OUTPUT

```
Ex-5 U=83.33% (C1=1, C2=1, C3=2; T1=2, T2=5, T3=15; T=D):
CT test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=5.000000, utility_sum = 0.700000
for 2, wcet=2.000000, period=15.000000, utility_sum = 0.833333
utility_sum = 0.833333
LUB = 0.779763
RM LUB INFEASIBLE
```

FEASIBILITY CODE – SCHEDULING POINT TEST OUTPUT

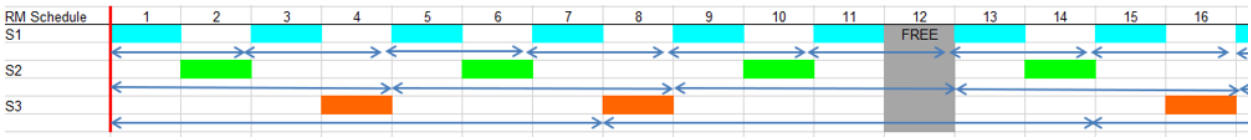
```
Ex-5 U=83.33% (C1=1, C2=1, C3=2; T1=2, T2=5, T3=15; T=D):
SP test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=5.000000, utility_sum = 0.700000
for 2, wcet=2.000000, period=15.000000, utility_sum = 0.833333
utility_sum = 0.833333
LUB = 0.779763
RM LUB INFEASIBLE
```

As per the code, the RM policy is feasible based on the feasibility tests. However, it does not satisfy the RM LUB condition and hence deems the scheduling policy infeasible. Hence the code output is partially in agreement with our analysis.

EXAMPLE 6:

Example 4	Services	Freq f	f0 multiple	Period		WCET		Utility					
	S1	0.5	3.5	T1	2	C1	1	U1	0.5	LCM =	28		
	S2	0.25	1.75	T2	4	C2	1	U2	0.25	LUB =	77.98%		
	S3	0.142857	1	T3	7	C3	1	U3	0.142857	Utot =	89.29%	10.71%	100.00%

RATE MONOTONIC SCHEDULING POLICY TIMING DIAGRAM



By observing the rate monotonic timing diagram, it can be concluded that the RM policy is schedulable without missing any deadlines.

CHEDDAR TOOL RATE MONOTONIC POLICY OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  22
- Number of preemptions :  0

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 2/worst
   S2 => 4/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 28.0, see Leung and Merill (1980) from [21].
- 3 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 0.89286 (see [1], page 6).
- Processor utilization factor with period is 0.89286 (see [1], page 6).
- In the preemptive case, with RM, we cannot prove that the task set is schedulable because the processor utilization factor 0.89286 is more than 0.77976


2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :  (see [2], page 3, equation 4).
   S0 => 1
   S1 => 2
   S2 => 4
- All task deadlines will be met : the task set is schedulable.
```

As expected from the tool output, the RM policy is schedulable which is the conclusion derived from the timing diagram. The cheddar tool feasibility test is done based on processor utilization as well as worst case timing response. Although the RM LUB condition is not satisfied, the schedule is feasible based on the timing analysis.

FEASIBILITY CODE – COMPLETION TEST OUTPUT

```
Ex-6 U=89.29% (C1=1, C2=1, C3=1; T1=2, T2=4, T3=7; T=D):
CT test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=4.000000, utility_sum = 0.750000
for 2, wcet=1.000000, period=7.000000, utility_sum = 0.892857
utility_sum = 0.892857
LUB = 0.779763
RM LUB INFEASIBLE
```

FEASIBILITY CODE – SCHEDULING POINT TEST OUTPUT

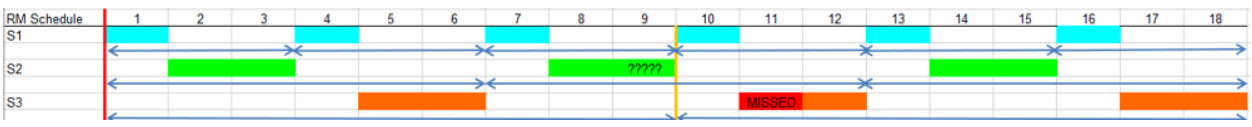
```
Ex-6 U=89.29% (C1=1, C2=1, C3=1; T1=2, T2=4, T3=7; T=D):
SP test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=4.000000, utility_sum = 0.750000
for 2, wcet=1.000000, period=7.000000, utility_sum = 0.892857
utility_sum = 0.892857
LUB = 0.779763
RM LUB INFEASIBLE
```

As per the code, the RM policy is feasible based on the feasibility tests. However, it does not satisfy the RM LUB condition and hence deems the scheduling policy infeasible. Hence the code output is partially in agreement with our analysis.

EXAMPLE 7:

Example 11	Service	Period		Freq f	f0 multiple	WCET		Utility			
	S1	T1	3	0.333333	3	C1	1	U1	33.33%	LCM =	18
	S2	T2	6	0.166667	1.5	C2	2	U2	33.33%	LUB =	77.98%
	S3	T3	9	0.111111	1	C3	3	U3	33.33%	Utot =	100.00%

RATE MONOTONIC SCHEDULING POLICY TIMING DIAGRAM



For the given set of services, according to the timing diagram analysis, Rate Monotonic policy is not feasible as it misses a few deadlines.

CHEDDAR TOOL RATE MONOTONIC POLICY OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  11
- Number of preemptions :  2

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 3/worst
   S2 => 11/worst ,  missed its deadline (absolute deadline =  9 ; completion time =  11)
- Some task deadlines will be missed : the task set is not schedulable.



Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 18.0, see Leung and Merill (1980) from [21].
- 0 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with RM, we cannot prove that the task set is schedulable because the processor utilization factor 1.00000 is more than 0.77976 (

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :  (see [2], page 3, equation 4).
   S0 => 1
   S1 => 3
   S2 => 11,  missed its deadline (deadline =  9)
- Some task deadlines will be missed : the task set is not schedulable.
```

The cheddar tool output is correct as expected. It gives an analysis of when the service deadline will be missed. Based on this the feasibility test is carried out. For this service set, the RM policy is not feasible.

FEASIBILITY CODE – COMPLETION TEST OUTPUT

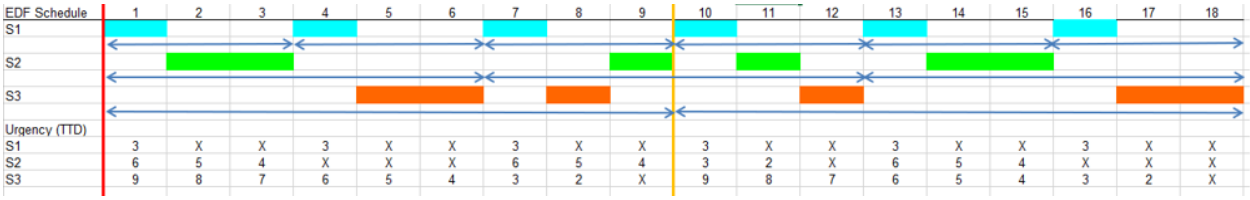
```
Ex-7 U=100.00% (C1=1, C2=2, C3=3; T1=3, T2=6, T3=9; T=D):
CT test INFEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=3.000000, utility_sum = 0.333333
for 1, wcet=2.000000, period=6.000000, utility_sum = 0.666667
for 2, wcet=3.000000, period=9.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.779763
RM LUB INFEASIBLE
```

FEASIBILITY CODE – SCHEDULING POINT TEST OUTPUT

```
Ex-7 U=100.00% (C1=1, C2=2, C3=3; T1=3, T2=6, T3=9; T=D):
SP test INFEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=3.000000, utility_sum = 0.333333
for 1, wcet=2.000000, period=6.000000, utility_sum = 0.666667
for 2, wcet=3.000000, period=9.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.779763
RM LUB INFEASIBLE
```

Similar to the output from timing analysis and cheddar tool, the feasibility code carries out the tests and checks RM LUB condition. As proven by the other tools, the RM policy is not feasible.

EARLIEST DEADLINE FIRST TIMING DIAGRAM



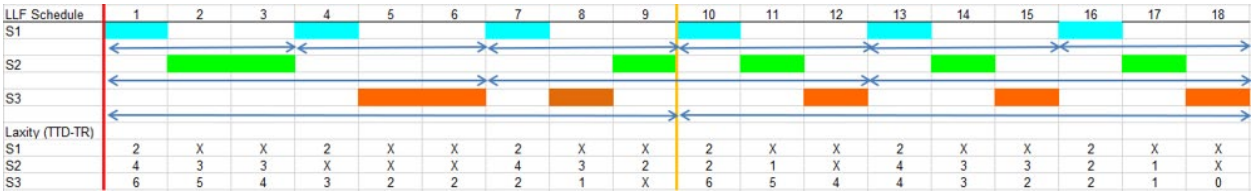
According to the timing diagram, using dynamic priority scheduling policy, the deadlines are not missed, and the schedule is feasible.

CHEDDAR TOOL EARLIEST DEADLINE FIRST OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  13
- Number of preemptions :  3

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 5/worst
   S2 => 9/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 18.0, see Goossens and Devillers (1997) from [21].
- 0 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (s


2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :
   S0 => 3
   S1 => 6
   S2 => 9
- All task deadlines will be met : the task set is schedulable.
```

Similar to timing analysis, the cheddar tool tests the service test feasibility using processor utilization factor and the worst-case response time analysis. The simulation and feasibility test proves that EDF policy is feasible for the given services.

LEAST LAXITY FIRST TIMING DIAGRAM



By observing the timing diagram for LLF, we can conclude that the LLF policy is feasible.

CHEDDAR TOOL LEAST LAXITY FIRST OUTPUT

Scheduling simulation, Processor CPU0 :
- Number of context switches : 11
- Number of preemptions : 2

- Task response time computed from simulation :
  S0 => 1/worst
  S1 => 3/worst
  S2 => 11/worst , missed its deadline (absolute deadline = 9 ; completion time = 11)
- Some task deadlines will be missed : the task set is not schedulable.

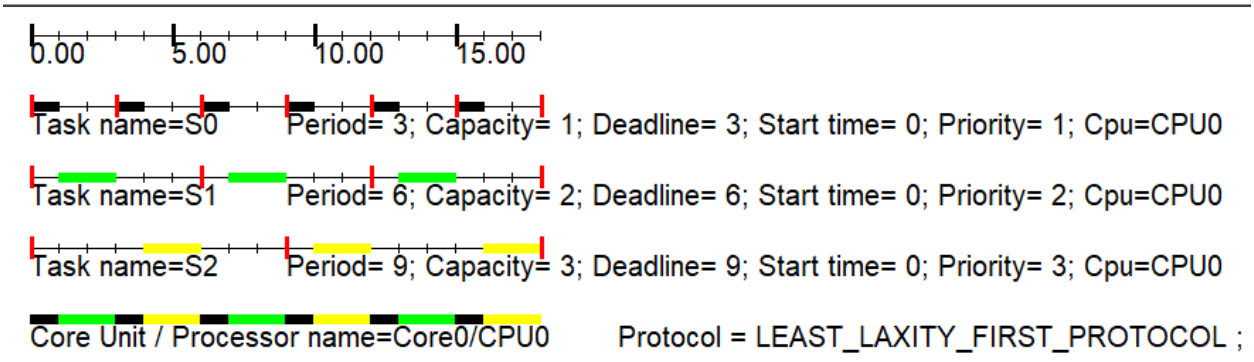
Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 18.0, see Leung and Merrill (1980) from [21].
- 0 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor : 1.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :
  S0 => 3
  S1 => 6
  S2 => 9
- All task deadlines will be met : the task set is schedulable.

The cheddar tool analysis gives the same output as timing diagram i.e., the policy is feasible. However, there is a glitch in the tool output analysis. The scheduling policy simulation result indicates that a few deadlines are missed which is clearly not the case.



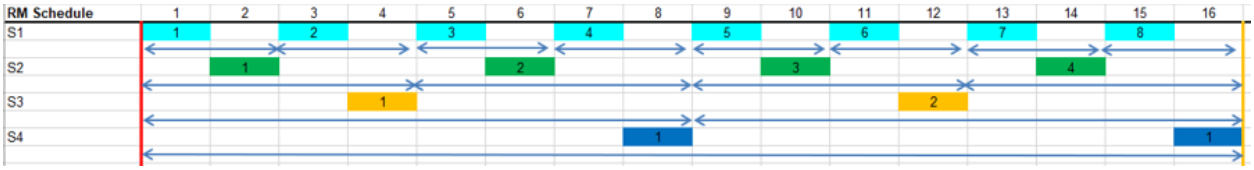
As we can observe in the above screenshot of timing analysis done by cheddar tool for LLF policy, there are a few mistakes in the diagram. Instead of LLF policy, the

diagram resembles the timing diagram of RM policy. Hence, the timing diagram analysis is not accurate for LLF in cheddar.

EXAMPLE 8:

Example Harmonic	Service	Freq f	f0 multiple	Period T							
	S1	0.5	8	T1	2	C1	1	U1	50.00%	LCM =	16
	S2	0.25	4	T2	4	C2	1	U2	25.00%		
	S3	0.125	2	T3	8	C3	1	U3	12.50%	LUB =	75.68%
	S4 (f0)	0.0625	1	T4	16	C4	2	U4	12.50%	U total =	100.00%

RATE MONOTONIC SCHEDULING POLICY TIMING DIAGRAM



By observing the rate monotonic timing diagram, it can be concluded that the RM policy is schedulable without missing any deadlines.

CHEDDAR TOOL RATE MONOTONIC POLICY OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  15
- Number of preemptions :  1

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 2/worst
   S2 => 4/worst
   S3 => 16/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.



Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 16.0, see Leung and Merill (1980) from [21].
- 0 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with RM, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000


2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :  (see [2], page 3, equation 4).
   S0 => 1
   S1 => 2
   S2 => 4
   S3 => 16
- All task deadlines will be met : the task set is schedulable.
```

As expected from the tool output, the RM policy is schedulable which is the conclusion derived from the timing diagram. The cheddar tool feasibility test is done based on processor utilization as well as worst case timing response. However, the feasibility test of cheddar is not accurate. The RM LUB computation done to check processor utilization is incorrect (actual RM LUB is 0.7568). Although the RM LUB condition is not satisfied, the feasibility test of cheddar gives misleading output. This analysis was done by hand evaluation of RM policy.

FEASIBILITY CODE – COMPLETION TEST OUTPUT

```
Ex-8 U=100.00% (C1=1, C2=1, C3=1, C4=2; T1=2, T2=4, T3=8, T4=16; T=D):
CT test FEASIBLE
for 4, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=4.000000, utility_sum = 0.750000
for 2, wcet=1.000000, period=8.000000, utility_sum = 0.875000
for 3, wcet=2.000000, period=16.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.756828
RM LUB INFEASIBLE
```

FEASIBILITY CODE – SCHEDULING POINT TEST OUTPUT

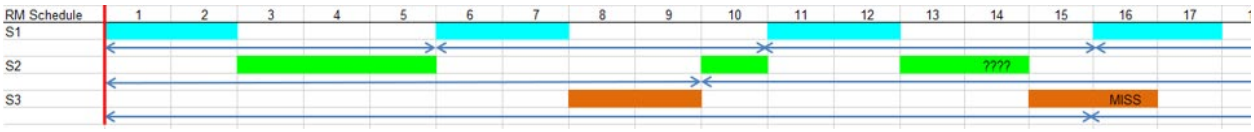
```
Ex-8 U=100.00% (C1=1, C2=1, C3=1, C4=2; T1=2, T2=4, T3=8, T4=16; T=D):
SP test FEASIBLE
for 4, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=4.000000, utility_sum = 0.750000
for 2, wcet=1.000000, period=8.000000, utility_sum = 0.875000
for 3, wcet=2.000000, period=16.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.756828
RM LUB INFEASIBLE
```

As per the code, the RM policy is feasible based on the feasibility tests. However, it does not satisfy the RM LUB condition and hence deems the scheduling policy infeasible. Hence the code output is partially in agreement with our analysis.

EXAMPLE 9:

Example 15	Service	Freq f	f0 multiple	Period		WCET		Utility			
	S1	0.2	3	T1	5	C1	2	U1	40.00%	LCM =	45
	S2	0.111111	1.666667	T2	9	C2	3	U2	33.33%	LUB =	77.98%
	S3	0.066667	1	T3	15	C3	4	U3	26.67%	Utot =	100.00%

RATE MONOTONIC SCHEDULING POLICY TIMING DIAGRAM



For the given set of services, according to the timing diagram analysis, Rate Monotonic policy is not feasible as it misses a few deadlines.

CHEDDAR TOOL RATE MONOTONIC POLICY OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  21
- Number of preemptions :  6

- Task response time computed from simulation :
   S0 => 2/worst
   S1 => 5/worst
   S2 => 19/worst ,  missed its deadline (absolute deadline =  15 ; completion time =  18),  missed its deadline (absolute deadline =  30 ; completion time =  34)
- Some task deadlines will be missed : the task set is not schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 45.0, see Leung and Merill (1980) from [21].
- 0 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with RM, we cannot prove that the task set is schedulable because the processor utilization factor 1.00000 is more than 0.77976 (see [1],


2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :  (see [2], page 3, equation 4).
   S0 => 2
   S1 => 5
   S2 => 19,  missed its deadline (deadline =  15)
- Some task deadlines will be missed : the task set is not schedulable.
```

The cheddar tool output is correct as expected. It gives an analysis of when the service deadline will be missed. Based on this the feasibility test is carried out. For this service set, the RM policy is not feasible.

FEASIBILITY CODE – COMPLETION TEST OUTPUT

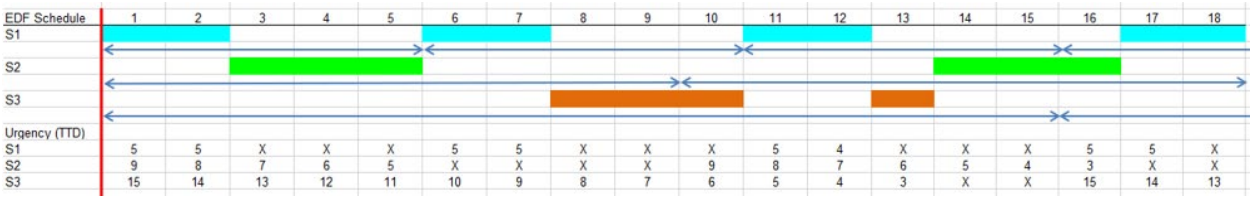
```
Ex-9 U=100.00% (C1=2, C2=3, C3=4; T1=5, T2=9, T3=15; T=D):
CT test INFEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=2.000000, period=5.000000, utility_sum = 0.400000
for 1, wcet=3.000000, period=9.000000, utility_sum = 0.733333
for 2, wcet=4.000000, period=15.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.779763
RM LUB INFEASIBLE
```

FEASIBILITY CODE – SCHEDULING POINT TEST OUTPUT

```
Ex-9 U=100.00% (C1=2, C2=3, C3=4; T1=5, T2=9, T3=15; T=D):
SP test INFEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=2.000000, period=5.000000, utility_sum = 0.400000
for 1, wcet=3.000000, period=9.000000, utility_sum = 0.733333
for 2, wcet=4.000000, period=15.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.779763
RM LUB INFEASIBLE
```

Similar to the output from timing analysis and cheddar tool, the feasibility code carries out the tests and checks RM LUB condition. As proven by the other tools, the RM policy is not feasible.

EARLIEST DEADLINE FIRST TIMING DIAGRAM



According to the timing diagram, using dynamic priority scheduling policy, the deadlines are not missed, and the schedule is feasible.

CHEDDAR TOOL EARLIEST DEADLINE FIRST OUTPUT

Scheduling simulation, Processor CPU0 :
- Number of context switches : 21
- Number of preemptions : 5

- Task response time computed from simulation :
  S0 => 3/worst
  S1 => 7/worst
  S2 => 15/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.

Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

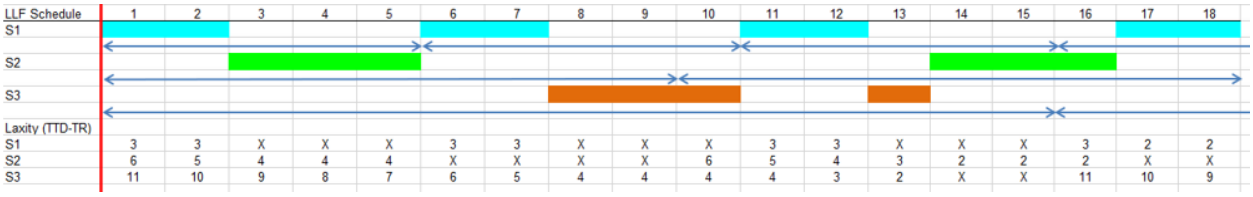
- The feasibility interval is 45.0, see Goossens and Devillers (1997) from [21].
- 0 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor : 1.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :
  S0 => 5
  S1 => 9
  S2 => 15
- All task deadlines will be met : the task set is schedulable.

Similar to timing analysis, the cheddar tool tests the service test feasibility using processor utilization factor and the worst-case response time analysis. The simulation and feasibility test proves that EDF policy is feasible for the given services.

LEAST LAXITY FIRST TIMING DIAGRAM

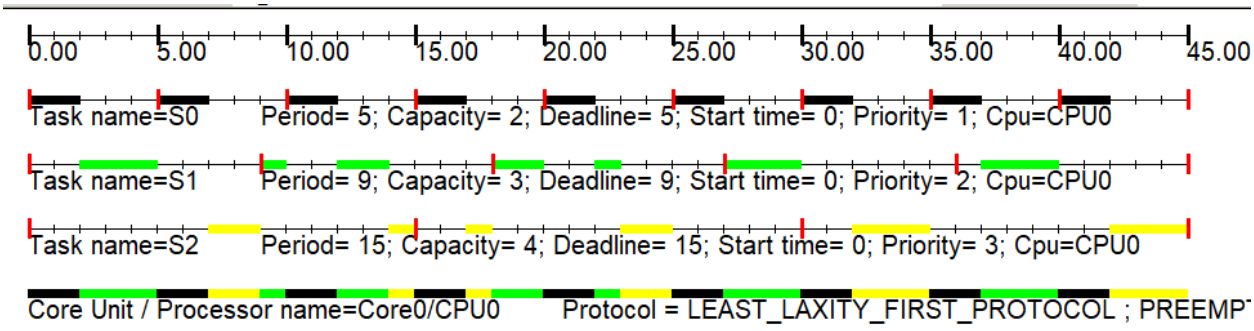


By observing the timing diagram for LLF, we can conclude that the LLF policy is feasible.

CHEDDAR TOOL LEAST LAXITY FIRST OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  21
- Number of preemptions :  6

- Task response time computed from simulation :
   S0 => 2/worst
   S1 => 5/worst
   S2 => 19/worst ,  missed its deadline (absolute deadline =  15 ; completion time =  18),  missed its deadline (absolute deadline =  30 ; completion time =  34)
- Some task deadlines will be missed : the task set is not schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 45.0, see Leung and Merrill (1980) from [21].
- 0 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (see [7]).

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :
   S0 => 5
   S1 => 9
   S2 => 15
- All task deadlines will be met : the task set is schedulable.
```

The cheddar tool analysis gives the same output as timing diagram i.e., the policy is feasible. However, there is a glitch in the tool output analysis. The scheduling policy simulation result indicates that a few deadlines are missed which is clearly not the case.

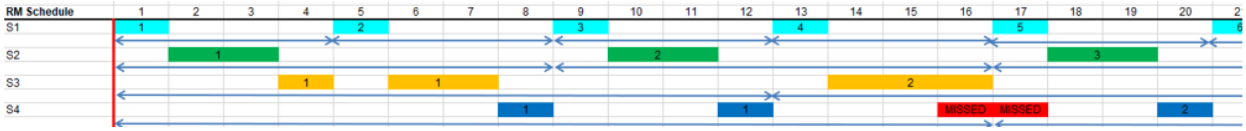


As we can observe in the above screenshot of timing analysis done by cheddar tool for LLF policy, there are a few mistakes in the diagram. Instead of LLF policy, the diagram resembles the timing diagram of RM policy. Hence, the timing diagram analysis is not accurate for LLF in cheddar.

EXAMPLE 10:

Example Butazzo	Service	Freq f	f0 multiple	Period T								LCM/T
	S1	0.25	4	T1	4	C1	1	U1	0.25	LCM =	48	12
	S2	0.125	2	T2	8	C2	2	U2	0.25			6
	S3	0.08333	1.33333	T3	12	C3	3	U3	0.25	LUB =	75.68%	4
	S4 (f0)	0.0625	1	T4	16	C4	4	U4	0.25	U total =	100.00%	3

RATE MONOTONIC SCHEDULING POLICY TIMING DIAGRAM



For the given set of services, according to the timing diagram analysis, Rate Monotonic policy is not feasible as it misses a few deadlines.

CHEDDAR TOOL RATE MONOTONIC POLICY OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  31
- Number of preemptions :  8

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 3/worst
   S2 => 7/worst
   S3 => 22/worst ,  missed its deadline (absolute deadline =  16 ; completion time =  22),  missed its deadline (absolute deadline =  32 ; completion time =  36)
- Some task deadlines will be missed : the task set is not schedulable.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 48.0, see Leung and Merill (1980) from [21].
- 0 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with RM, we cannot prove that the task set is schedulable because the processor utilization factor 1.00000 is more than 0.75683 (see [1],


2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :  (see [2], page 3, equation 4).
   S0 => 1
   S1 => 3
   S2 => 7
   S3 => 22,  missed its deadline (deadline =  16)
- Some task deadlines will be missed : the task set is not schedulable.
```

The cheddar tool output is correct as expected. It gives an analysis of when the service deadline will be missed. Based on this the feasibility test is carried out. For this service set, the RM policy is not feasible.

FEASIBILITY CODE – COMPLETION TEST OUTPUT

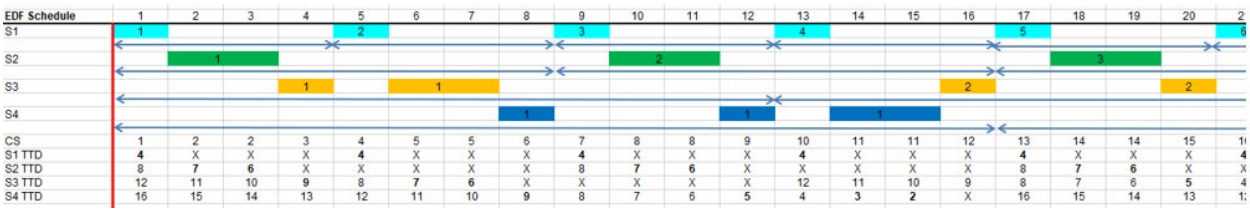
```
Ex-10 U=100.00% (C1=1, C2=2, C3=3, C4=4; T1=4, T2=8, T3=12, T4=16; T=D):
CT test INFEASIBLE
for 4, utility_sum = 0.000000
for 0, wcet=1.000000, period=4.000000, utility_sum = 0.250000
for 1, wcet=2.000000, period=8.000000, utility_sum = 0.500000
for 2, wcet=3.000000, period=12.000000, utility_sum = 0.750000
for 3, wcet=4.000000, period=16.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.756828
RM LUB INFEASIBLE
```

FEASIBILITY CODE – SCHEDULING POINT TEST OUTPUT

```
Ex-10 U=100.00% (C1=1, C2=2, C3=3, C4=4; T1=4, T2=8, T3=12, T4=16; T=D):
SP test INFEASIBLE
for 4, utility_sum = 0.000000
for 0, wcet=1.000000, period=4.000000, utility_sum = 0.250000
for 1, wcet=2.000000, period=8.000000, utility_sum = 0.500000
for 2, wcet=3.000000, period=12.000000, utility_sum = 0.750000
for 3, wcet=4.000000, period=16.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.756828
RM LUB INFEASIBLE
```

Similar to the output from timing analysis and cheddar tool, the feasibility code carries out the tests and checks RM LUB condition. As proven by the other tools, the RM policy is not feasible.

EARLIEST DEADLINE FIRST TIMING DIAGRAM



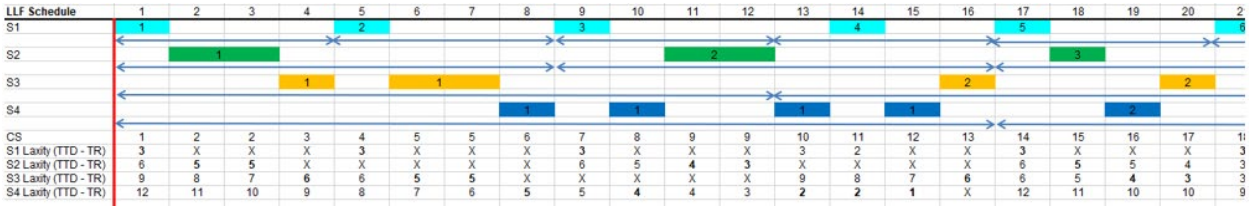
According to the timing diagram, using dynamic priority scheduling policy, the deadlines are not missed, and the schedule is feasible.

CHEDDAR TOOL EARLIEST DEADLINE FIRST OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  33
- Number of preemptions :  9

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 4/worst
   S2 => 10/worst
   S3 => 16/worst
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.


Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 48.0, see Goossens and Devillers (1997) from [21].
- 0 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with EDF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000


2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :
   S0 => 4
   S1 => 8
   S2 => 12
   S3 => 16
- All task deadlines will be met : the task set is schedulable.
```

Similar to timing analysis, the cheddar tool tests the service test feasibility using processor utilization factor and the worst-case response time analysis. The simulation and feasibility test proves that EDF policy is feasible for the given services.

LEAST LAXITY FIRST TIMING DIAGRAM

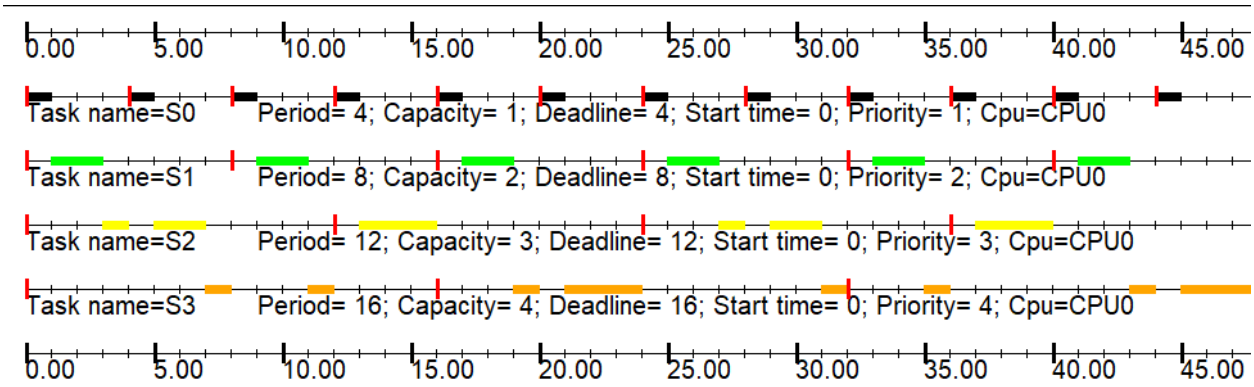


By observing the timing diagram for LLF, we can conclude that the LLF policy is feasible.

CHEDDAR TOOL LEAST LAXITY FIRST OUTPUT

```
Scheduling simulation, Processor CPU0 :
- Number of context switches :  31
- Number of preemptions :  8

- Task response time computed from simulation :
   S0 => 1/worst
   S1 => 3/worst
   S2 => 7/worst
   S3 => 22/worst ,  missed its deadline (absolute deadline =  16 ; completion time =  22),  missed its deadline (absolute deadline =  32 ; completion time =  36)
- Some task deadlines will be missed : the task set is not schedulable.



Scheduling feasibility, Processor CPU0 :
1) Feasibility test based on the processor utilization factor :

- The feasibility interval is 48.0, see Leung and Merrill (1980) from [21].
- 0 units of time are unused in the feasibility interval.
- Number of cores hosted by this processor :  1.
- Processor utilization factor with deadline is 1.00000 (see [1], page 6).
- Processor utilization factor with period is 1.00000 (see [1], page 6).
- In the preemptive case, with LLF, the task set is schedulable because the processor utilization factor 1.00000 is equal or less than 1.00000 (see [7]).

2) Feasibility test based on worst case response time for periodic tasks :

- Worst case task response time :
   S0 => 4
   S1 => 8
   S2 => 12
   S3 => 16
- All task deadlines will be met : the task set is schedulable.
```

The cheddar tool analysis gives the same output as timing diagram i.e., the policy is feasible. However, there is a glitch in the tool output analysis. The scheduling policy simulation result indicates that a few deadlines are missed which is clearly not the case.



As we can observe in the above screenshot of timing analysis done by cheddar tool for LLF policy, there are a few mistakes in the diagram. Instead of LLF policy, the diagram resembles the timing diagram of RM policy. Hence, the timing diagram analysis is not accurate for LLF in cheddar.

c. Does your modified Feasibility code agree with Cheddar analysis in all 5 additional cases? Why or why not?

I chose 6 random examples to check the accuracy of cheddar tool and the feasibility code. To analyze the feasibility code, I added the period and worst-case execution time for all the examples in the code. These timings are arrays for the service sets.

```
// U=0.83
U32_T ex5_period[] = {2, 5, 15};
U32_T ex5_wcet[] = {1, 1, 2};

// U=0.89
U32_T ex6_period[] = {2, 4, 7};
U32_T ex6_wcet[] = {1, 1, 1};

// U=1.00
U32_T ex7_period[] = {3, 6 ,9};
U32_T ex7_wcet[] = {1, 2, 3};

// U=1.00
U32_T ex8_period[] = {2, 4, 8, 16};
U32_T ex8_wcet[] = {1, 1, 1, 2};

// U=1.00
U32_T ex9_period[] = {5, 9, 15};
U32_T ex9_wcet[] = {2, 3, 4};

// U=1.00
U32_T ex10_period[] = {4, 8, 12, 16};
U32_T ex10_wcet[] = {1, 2, 3, 4};
```

Further, I added different cases of the service sets for completion test, scheduling point test and RM LUB calculation. The if…else clauses call the feasibility test functions and give the test output based on the function return value.

```
printf("Ex-5 U=%4.2f\% (C1=1, C2=1, C3=2; T1=2, T2=5, T3=15; T=D): ",
               ((1.0/2.0)*100.0 + (1.0/5.0)*100.0 + (2.0/15.0)*100.0));
    numServices = 3;
if(completion_time_feasibility(numServices, ex5_period, ex5_wcet, ex5_period) == TRUE)
    printf("\nCT test FEASIBLE\n");
else
    printf("\nCT test INFEASIBLE\n");

if(rate_monotonic_least_upper_bound(numServices, ex5_period, ex5_wcet, ex5_period) == TRUE)
    printf("RM LUB FEASIBLE\n");
else
    printf("RM LUB INFEASIBLE\n");
printf("\n");

printf("Ex-6 U=%4.2f\% (C1=1, C2=1, C3=1; T1=2, T2=4, T3=7; T=D): ",
               ((1.0/2.0)*100.0 + (1.0/4.0)*100.0 + (1.0/7.0)*100.0));
    numServices = 3;
if(completion_time_feasibility(numServices, ex6_period, ex6_wcet, ex6_period) == TRUE)
    printf("\nCT test FEASIBLE\n");
else
    printf("\nCT test INFEASIBLE\n");

if(rate_monotonic_least_upper_bound(numServices, ex6_period, ex6_wcet, ex6_period) == TRUE)
    printf("RM LUB FEASIBLE\n");
else
    printf("RM LUB INFEASIBLE\n");
printf("\n");

printf("Ex-7 U=%4.2f\% (C1=1, C2=2, C3=3; T1=3, T2=6, T3=9; T=D): ",
               ((1.0/3.0)*100.0 + (2.0/6.0)*100.0 + (3.0/9.0)*100.0));
    numServices = 3;
if(completion_time_feasibility(numServices, ex7_period, ex7_wcet, ex7_period) == TRUE)
    printf("\nCT test FEASIBLE\n");
else
    printf("\nCT test INFEASIBLE\n");

if(rate_monotonic_least_upper_bound(numServices, ex7_period, ex7_wcet, ex7_period) == TRUE)
    printf("RM LUB FEASIBLE\n");
else
    printf("RM LUB INFEASIBLE\n");
printf("\n");

printf("Ex-8 U=%4.2f\% (C1=1, C2=1, C3=1, C4=2; T1=2, T2=4, T3=8, T4=16; T=D): ",
               ((1.0/2.0)*100.0 + (1.0/4.0)*100.0 + (1.0/8.0)*100.0 + (2.0/16.0)*100.0));
    numServices = 4;
if(completion_time_feasibility(numServices, ex8_period, ex8_wcet, ex8_period) == TRUE)
    printf("\nCT test FEASIBLE\n");
```

```
printf("Ex-5 U=%4.2f\% (C1=1, C2=1, C3=2; T1=2, T2=5, T3=15; T=D): ",
               ((1.0/2.0)*100.0 + (1.0/5.0)*100.0 + (2.0/15.0)*100.0));
    numServices = 3;
if(scheduling_point_feasibility(numServices, ex5_period, ex5_wcet, ex5_period) == TRUE)
    printf("\nSP test FEASIBLE\n");
else
    printf("\nSP test INFEASIBLE\n");

if(rate_monotonic_least_upper_bound(numServices, ex5_period, ex5_wcet, ex5_period) == TRUE)
    printf("RM LUB FEASIBLE\n");
else
    printf("RM LUB INFEASIBLE\n");
printf("\n");

printf("Ex-6 U=%4.2f\% (C1=1, C2=1, C3=1; T1=2, T2=4, T3=7; T=D): ",
               ((1.0/2.0)*100.0 + (1.0/4.0)*100.0 + (1.0/7.0)*100.0));
    numServices = 3;
if(scheduling_point_feasibility(numServices, ex6_period, ex6_wcet, ex6_period) == TRUE)
    printf("\nSP test FEASIBLE\n");
else
    printf("\nSP test INFEASIBLE\n");

if(rate_monotonic_least_upper_bound(numServices, ex6_period, ex6_wcet, ex6_period) == TRUE)
    printf("RM LUB FEASIBLE\n");
else
    printf("RM LUB INFEASIBLE\n");
printf("\n");

printf("Ex-7 U=%4.2f\% (C1=1, C2=2, C3=3; T1=3, T2=6, T3=9; T=D): ",
               ((1.0/3.0)*100.0 + (2.0/6.0)*100.0 + (3.0/9.0)*100.0));
    numServices = 3;
if(scheduling_point_feasibility(numServices, ex7_period, ex7_wcet, ex7_period) == TRUE)
    printf("\nSP test FEASIBLE\n");
else
    printf("\nSP test INFEASIBLE\n");

if(rate_monotonic_least_upper_bound(numServices, ex7_period, ex7_wcet, ex7_period) == TRUE)
    printf("RM LUB FEASIBLE\n");
else
    printf("RM LUB INFEASIBLE\n");
printf("\n");

printf("Ex-8 U=%4.2f\% (C1=1, C2=1, C3=1, C4=2; T1=2, T2=4, T3=8, T4=16; T=D): ",
```

Below is the output of the feasibility code after completion test and scheduling point test.

```
CT test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=4.000000, utility_sum = 0.750000
for 2, wcet=1.000000, period=7.000000, utility_sum = 0.892857
utility_sum = 0.892857
LUB = 0.779763
RM LUB INFEASIBLE

Ex-7 U=100.00% (C1=1, C2=2, C3=3; T1=3, T2=6, T3=9; T=D):
CT test INFEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=3.000000, utility_sum = 0.333333
for 1, wcet=2.000000, period=6.000000, utility_sum = 0.666667
for 2, wcet=3.000000, period=9.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.779763
RM LUB INFEASIBLE

Ex-8 U=100.00% (C1=1, C2=1, C3=1, C4=2; T1=2, T2=4, T3=8, T4=16; T=D):
CT test FEASIBLE
for 4, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=4.000000, utility_sum = 0.750000
for 2, wcet=1.000000, period=8.000000, utility_sum = 0.875000
for 3, wcet=2.000000, period=16.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.756828
RM LUB INFEASIBLE

Ex-9 U=100.00% (C1=2, C2=3, C3=4; T1=5, T2=9, T3=15; T=D):
CT test INFEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=2.000000, period=5.000000, utility_sum = 0.400000
for 1, wcet=3.000000, period=9.000000, utility_sum = 0.733333
for 2, wcet=4.000000, period=15.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.779763
RM LUB INFEASIBLE

Ex-10 U=100.00% (C1=1, C2=2, C3=3, C4=4; T1=4, T2=8, T3=12, T4=16; T=D):
CT test INFEASIBLE
for 4, utility_sum = 0.000000
for 0, wcet=1.000000, period=4.000000, utility_sum = 0.250000
for 1, wcet=2.000000, period=8.000000, utility_sum = 0.500000
for 2, wcet=3.000000, period=12.000000, utility_sum = 0.750000
for 3, wcet=4.000000, period=16.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.756828
RM LUB INFEASIBLE
```

```
Ex-6 U=89.29% (C1=1, C2=1, C3=1; T1=2, T2=4, T3=7; T=D):
SP test FEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=4.000000, utility_sum = 0.750000
for 2, wcet=1.000000, period=7.000000, utility_sum = 0.892857
utility_sum = 0.892857
LUB = 0.779763
RM LUB INFEASIBLE

Ex-7 U=100.00% (C1=1, C2=2, C3=3; T1=3, T2=6, T3=9; T=D):
SP test INFEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=1.000000, period=3.000000, utility_sum = 0.333333
for 1, wcet=2.000000, period=6.000000, utility_sum = 0.666667
for 2, wcet=3.000000, period=9.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.779763
RM LUB INFEASIBLE

Ex-8 U=100.00% (C1=1, C2=1, C3=1, C4=2; T1=2, T2=4, T3=8, T4=16; T=D):
SP test FEASIBLE
for 4, utility_sum = 0.000000
for 0, wcet=1.000000, period=2.000000, utility_sum = 0.500000
for 1, wcet=1.000000, period=4.000000, utility_sum = 0.750000
for 2, wcet=1.000000, period=8.000000, utility_sum = 0.875000
for 3, wcet=2.000000, period=16.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.756828
RM LUB INFEASIBLE

Ex-9 U=100.00% (C1=2, C2=3, C3=4; T1=5, T2=9, T3=15; T=D):
SP test INFEASIBLE
for 3, utility_sum = 0.000000
for 0, wcet=2.000000, period=5.000000, utility_sum = 0.400000
for 1, wcet=3.000000, period=9.000000, utility_sum = 0.733333
for 2, wcet=4.000000, period=15.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.779763
RM LUB INFEASIBLE

Ex-10 U=100.00% (C1=1, C2=2, C3=3, C4=4; T1=4, T2=8, T3=12, T4=16; T=D):
SP test INFEASIBLE
for 4, utility_sum = 0.000000
for 0, wcet=1.000000, period=4.000000, utility_sum = 0.250000
for 1, wcet=2.000000, period=8.000000, utility_sum = 0.500000
for 2, wcet=3.000000, period=12.000000, utility_sum = 0.750000
for 3, wcet=4.000000, period=16.000000, utility_sum = 1.000000
utility_sum = 1.000000
LUB = 0.756828
RM LUB INFEASIBLE
```

Feasibility code output analysis:

- The output of the feasibility code was mostly consistent with the output from cheddar tool and timing diagram analysis.
- For example 5, example 6 and example 8, the RM policy is feasible according to timing diagram, cheddar tool as well as feasibility tests carried out by the code. Cheddar tool tests the feasibility of service scheduling policy based on processor utilization (RM LUB) and timing diagram analysis. Sometimes even though RM LUB condition is not satisfied because of less CPU margin, the policy can still be feasible if it does not miss any deadlines. This analysis is done based on the timing diagram. Cheddar tool generates a timing diagram for the service sets and specified scheduling policy and checks for any deadline misses. However, the RM LUB condition was not satisfied due to its pessimistic testing. Also, the RM LUB value calculated by cheddar tool for example 8 was incorrect but the final output was correct.
- For example 7, 9 and 10, the RM policy is not feasible according to the timing diagram, cheddar tool output and the feasibility tests. For these examples, we can observe that the deadlines were missed for some services from timing diagram; the cheddar tool also correctly analyzed all deadline misses; moreover, these service sets did not pass the completion feasibility test and scheduling point feasibility test and also did not satisfy RM LUB condition. Hence, all the tools had the same analysis for these services.
- When for examples 7,9 and 10, RM policy was not feasible, I checked the Earliest Deadline First and Least Laxity First Policy feasibility. For all three examples, the EDF policy was feasible as per the timing diagram as well as cheddar tool output. Hence both the outputs were consistent. For LLF policy, the schedule was feasible according to both timing diagram provided to us and the cheddar tool. However, the timing diagram generated by cheddar tool was incorrect for LLF policy. Even though the final outcome showed that the schedule was feasible, initial simulation showed deadline misses which was not the case here.

Conclusion:

From the above analysis of all the test outcomes we can conclude that -

- RM LUB condition defined by Liu and Layland is not the accurate basis to check policy feasibility. It is a pessimistic approach which can consider a feasible schedule to be infeasible based on CPU utilization.
- Completion test and Scheduling point test are the best method to check RM policy feasibility as it would never approve an infeasible schedule but also approve all feasible schedules no matter what the processor utilization.
- Cheddar tool is an easy but inaccurate mechanism. The RM LUB calculation is sometimes incorrect and also the LLF policy timing diagram generation is not correct. It can be used for quick review of a policy but hand analysis should be considered for detailed scheduling policy analysis.

4. Read Chapter 3 of the textbook.

a. Provide 3 constraints that are made in the RM LUB derivation and 3 assumptions as documented in the Liu and Layland paper and in Chapter 3 of RTECS with Linux and RTOS.

Constraints made in RM LUB derivation:

- Constraint 1: The deadline for a task is equal to the period of the task. Hence, the task should be completed before another request for the task execution
- Constraint 2: Rate monotonic policy is a Fixed priority, preemptive, run-to-completion scheduling policy. This does not consider the necessity to run a higher priority task as an exception and not miss any deadline. Once the priorities are assigned, they cannot be changed while execution.
- Constraint 3: Liu and Layland have only talked about CPU sharing by the services and ignored other resource sharing like IO or memory; thus, making the RM derivation less complex and also less practical.

Assumptions made by Liu and Layland:

The most important assumptions made by Liu and Layland in their paper is assumption A1 and A4. They described these two assumptions as most important

and least defensible and that these assumptions should be made design goals for real-time systems to meet hard deadlines.

- Assumption (A1) - The requests for all tasks for which hard deadlines exist are periodic, with constant interval between requests.

- Assumption (A4) - Run-time for each task is constant for that task and does not vary with time. Run-time here refers to the time which is taken by a processor to execute the task without interruption.

According to Liu and Layland, if these two assumptions do not hold true, we would need detailed knowledge of run-time periods and request periods of the services. Unless these values are known, other analysis would also not work for scheduling.

- Assumption (A2) - Deadlines consist of run-ability constraints only--i.e., each task must be completed before the next request for it occurs. This assumption would eliminate all queuing problems but for it to work, we would need buffering hardware for each function.

b. Finally, list 3 key derivation steps in the RM LUB derivation that you either do not understand or that you would consider “tricky” math. Attempt to describe the rationale for those steps as best you can do based upon reading in Chapter 3 of RTECS with Linux and RTOS

On initial readings of the Liu and Layland paper, I was not able to understand the derivation for RM LUB but after a detailed explanation of that by Prof. Seiwert a few of those theorems and mathematical steps became clearer. Yet, there are a couple of points which I am not able to interpret.

Aspects I did not understand about Liu and Layland's fixed priority LUB derivation:

- In theorem 5, the use of polynomial equation and variables q and r was unclear to me. I could not relate the use of this change in equations with how we can derive LUB from there.

not by τ_i'. Let U' denote the utilization factor of such a set of tasks.

$$U' < U + [(q-1)C_i/T_m] + (C_i/T_i') - (C_i/T_i)$$

or

$$U' \leq U + C_i(q-1)[1/(qT_i + r) - (1/qT_i)].$$

Since $q - 1 > 0$ and $[1/(qT_i + r)] - (1/qT_i) \leq 0$, $U' \leq U$. The

- Similarly, for theorem 4 I was not able to understand the significance of C` and C`` in deriving LUB. Consequently, the derivation of equation for $C_m$ from C`, C``, T and U was very unclear to me. A proper explanation and detailed derivation with steps for these equations would have made the understanding procedure easier.

Let

$$\begin{aligned} C_1'' &= T_2 - T_1 \\ C_2'' &= C_2 - 2\Delta \\ C_3'' &= C_3 \\ &\vdots \\ C_{m-1}'' &= C_{m-1} \\ C_m'' &= C_m . \end{aligned}$$

Again, C_1'', C_2'', $\cdots$, C_{m-1}'', C_m'' fully utilize the processor. Let U'' denote the corresponding utilization factor. We have

$$U - U'' = -(\Delta/T_1) + (2\Delta/T_2) > 0.$$

Therefore, if indeed U is the minimum utilization factor, then

$$C_1 = T_2 - T_1$$

In a similar way, we can show that

$$\begin{aligned} C_2 &= T_3 - T_2 \\ C_3 &= T_4 - T_3 \\ &\vdots \\ C_{m-1} &= T_m - T_{m-1} . \end{aligned}$$

Tricky Mathematical Steps in the Derivation:

Apart from the above mentioned two theorems, I was able to understand all steps. The derivation done by Liu and Layland and assumptions considered for RM LUB from scratch is quite remarkable. One tricky mathematically baffling equation for me was the equation for run-time C1.

Case 1. The run-time C_1 is short enough that all requests for τ_1 within the critical time zone of T_2 are completed before the second τ_2 request. That is,

$$C_1 \leq T_2 - T_1 \lfloor T_2/T_1 \rfloor .$$

Case 2. The execution of the $\lceil T_2/T_1 \rceil$th request for t_1 overlaps the second request for τ_2. In this case

$$C_1 \geq T_2 - T_1 \lfloor T_2/T_1 \rfloor .$$

It was difficult for me to understand this part without any help. Here, Liu and Layland are trying to put the run-time in mathematical form which we normally observe in timing diagrams. They use two different cases for run-time to demonstrate complete CPU utilization and processor time.

Reference:

- REAL-TIME EMBEDDED COMPONENTS AND SYSTEMS with LINUX and RTOS by Dr. Sam Siewert and John Pratt
- Course Website - ecee.colorado.edu/~ecen5623/index_summer.html
- Architecture of the Space Shuttle Primary Avionics Software System paper by Gene Carlow
- Liu and LAyland Paper - Rate-Monotonic-Theory-Liu-and-Layland.pdf (colorado.edu)
- Cheddar Tool - Cheddar: an open-source real-time schedulability tool/scheduling simulator (univ-brest.fr)
- Cheddar documentation - singhoff04a.pdf (univ-brest.fr)
- Code, service set examples and timing diagrams – course website, provided by Prof. Seiwert.