Overview

We have introduced the first part of GRMS, the analysis of independent periodic tasks using the utilization bound test.

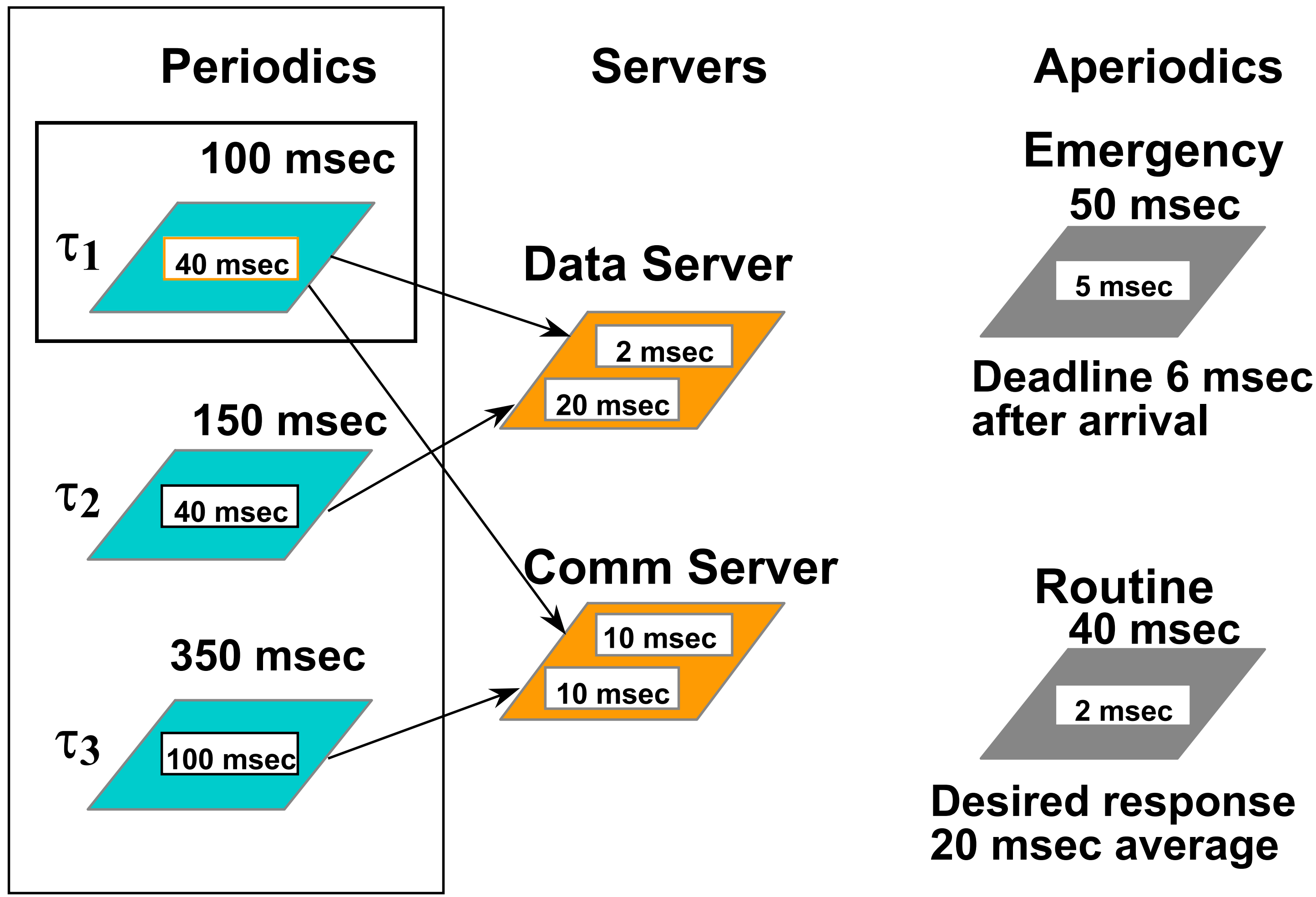
However, the utilization bound (UB) test is only a sufficient condition. When a set of periodic tasks fails the UB test, they may or may not be schedulable.

To resolve this ambiguity, the exact test was developed and we will study it in this lecture.

Review: Schedulability: UB Test

- Utilization bound(UB) test[Liu73]: a set of n independent periodic tasks scheduled by the rate monotonic algorithm will always meet its deadlines, for all task phasing, if
- $$\frac{C_1}{T_1} + \dots + \frac{C_n}{T_n} \leq U(n) = n(2^{1/n} - 1)$$
- U(1) = 1.0 U(4) = 0.756 U(7) = 0.728
- U(2) = 0.828 U(5) = 0.743 U(8) = 0.724
- U(3) = 0.779 U(6) = 0.734 U(9) = 0.720
- For harmonic task sets, the utilization bound is U(n)=1.00 for all n. For large n, the bound converges to *ln 2* ~ 0.69.
- Conventions, task 1 has shorter period than task 2 and so on.

Periodic Tasks



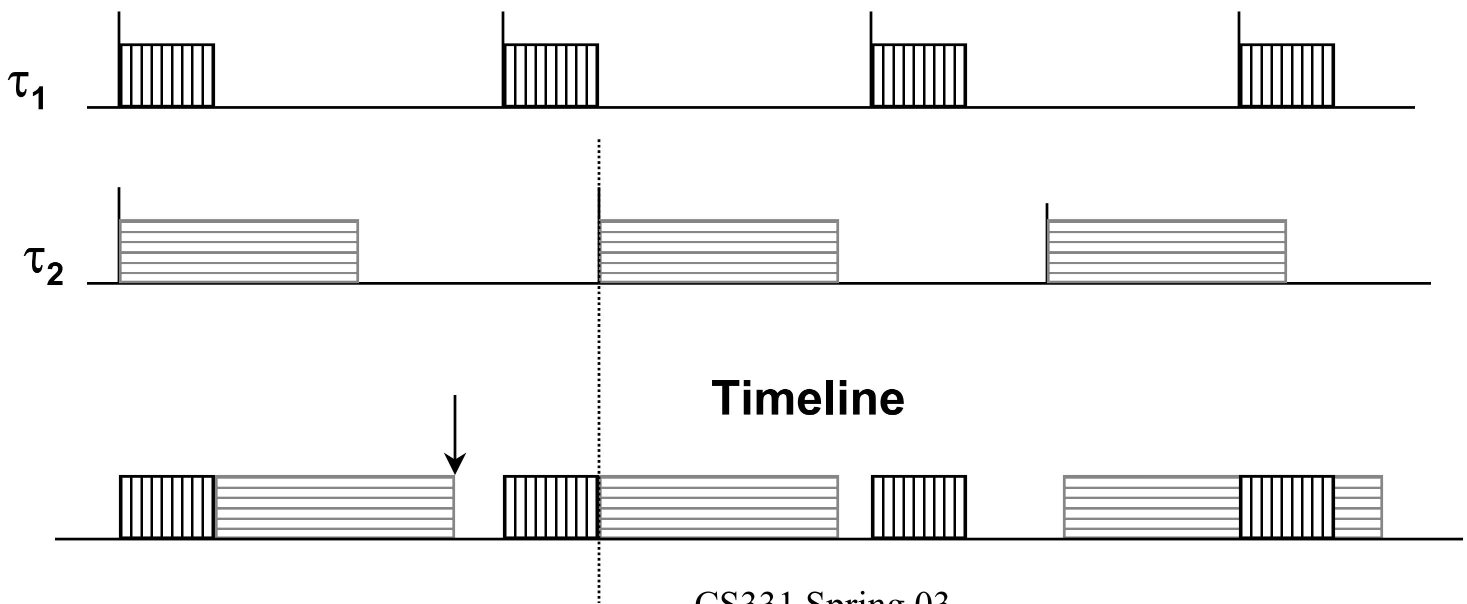
Sample Problem: Applying UB Test

	C	T	U
Task $\tau 1$:	**40**	**100**	**0.400**
Task $\tau 2$:	**40**	**150**	**0.267**
Task $\tau 3$:	**100**	**350**	**0.286**

- Utilization of first two tasks: 0.667 < U(2) = 0.828
  - first two tasks are schedulable by UB test
- Utilization of all three tasks: 0.953 > U(3) = 0.779
  - UB test is a sufficient condition and thus inconclusive
  - need to apply exact test.

The Exact Schedulability Test

Critical instant theorem: If a task meets its first deadline when all higher priority tasks are started at the same time, then this task's future deadlines will always be met. The exact test for a task checks if this task can meet its first deadline[Liu73].



Intuition for the Exact Schedulability Test

- Suppose we have n tasks, and we pick a task, say i, to see if it is schedulable.
- We initialize the testing by assuming all the higher priority tasks from 1 to i-1 will only preempt task i once.
- I.e. the initially presumed finishing time for task i is just the sum of C_1 to C_i , which we call a_0
- We now check the actual arrival of higher priority tasks within the duration a_0 and then presume that it will be all the preemption task i will experience. So we compute a_1 under this assumption.
- We will repeat this process until one of the two conditions occur:
  - 1. The a_n eventually exceeds the deadline of task i. In this case we terminate the iteration and conclude that task i is not schedulable.
  - 2. The series a_n converges to a fixed point (I.e. it stops increasing). If this fixed point is less than the deadline, then the task is schedulable and we terminate the iteration.

Exact Schedulability Test

- $$a_{n+1} = C_i + \sum_{j=1}^{i-1} \left\lceil \frac{a_n}{T_j} \right\rceil C_j \qquad \text{where } a_0 = \sum_{j=1}^{i} C_j$$

  Test terminates when $a_{n+1} > T_i$ (not schedulable)
  or when $a_{n+1} = a_n \leq T_i$ (schedulable).

The subscript to *a* indicates the number of iterations in the calculation.
The index *i* indicates it is the *ith* task being checked.

The index *j* runs from 1 to i-1, i.e all the higher priority tasks. Recall from the convention - task 1 has a higher priority than task 2 and so on.

Example: Applying Exact Test -2

- Use exact test to determine if τ_3 meets its first deadline:

$$a_0 = \sum_{j=1}^{3} C_j = C_1 + C_2 + C_3 = 40 + 40 + 100 = 180$$

$$a_1 = C_3 + \sum_{j=1}^{2} \left\lceil \frac{a_0}{T_j} \right\rceil C_j$$

$$= 100 + \left\lceil \frac{180}{100} \right\rceil (40) + \left\lceil \frac{180}{150} \right\rceil (40) = 100 + 80 + 80 = 260$$

Example: Applying the Exact Test -3

$$a_2 = C_3 + \sum_{j=1}^{2} \left\lceil \frac{a_1}{T_j} \right\rceil C_j = 100 + \left\lceil \frac{260}{100} \right\rceil (40) + \left\lceil \frac{260}{150} \right\rceil (40) = 300$$

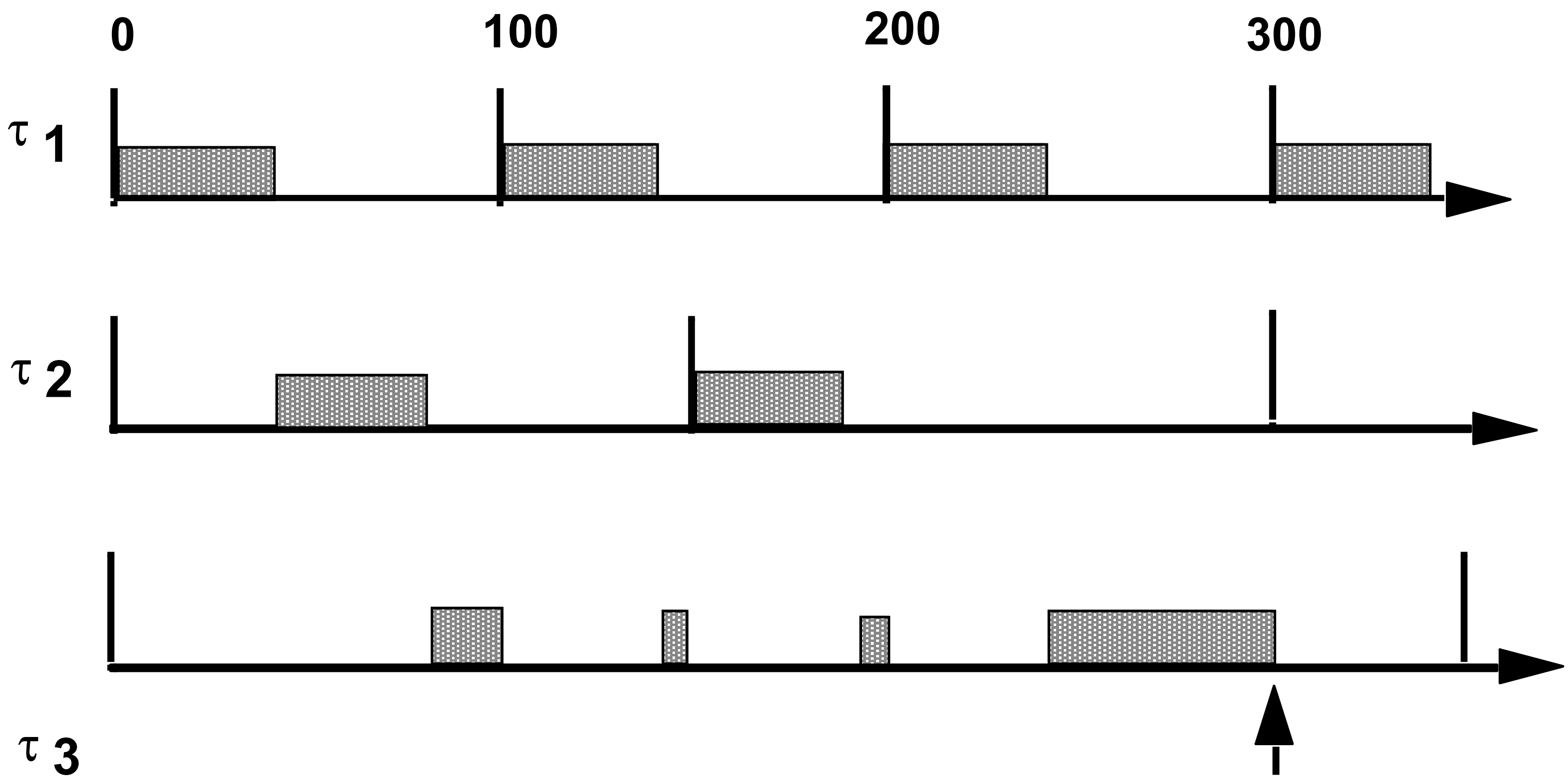
$$a_3 = C_3 + \sum_{j=1}^{2} \left\lceil \frac{a_2}{T_j} \right\rceil C_j = 100 + \left\lceil \frac{300}{100} \right\rceil (40) + \left\lceil \frac{300}{150} \right\rceil (40) = 300$$

$$a_3 = a_2 = 300 \quad \textbf{Done!}$$

- Task τ_3 is schedulable using exact test

$$a_3 = 300 < T = 350$$

Timeline



τ 3 completes its work at t = 300

Class Exercise

Suppose that we have three tasks

- $c1 = 4$, $T1 = 10$
- $c2 = 6.1$, $T2 = 14$

- Use the exact formula to show task 2 is not schedulable. Draw a timeline to confirm that.

- Can we add a task 3 with $C3 = 1$ and $T3 = 70$. What would be the shortest period of T3 that it can still meet its deadlines?

Class Exercise (continued)

$$a_0 = C_1 + C_2 + C_3 = \quad + \quad + \quad =$$

$$a_1 = C_3 + \sum_{j=1}^{2} \left\lceil \frac{a_0}{T_j} \right\rceil C_j = \quad + \left\lceil \frac{\quad}{\quad} \right\rceil () + \left\lceil \frac{\quad}{\quad} \right\rceil (\quad) =$$

$a_2 =$

$a_3 =$

Summary

We have now reviewed the UB test and exact test to determine if a set of independent periodic tasks are schedulable.

The schedulability analysis of independent periodic tasks are the foundation of GRMS and it is VERY IMPORTANT for you to master these test and can apply them correctly.