Real-Time Scheduling

Introduction to Real-Time

Review

- Main vocabulary
  - Definitions of tasks, task invocations, release/arrival time, absolute deadline, relative deadline, period, start time, finish time, ...
  - Preemptive versus non-preemptive scheduling
  - Priority-based scheduling
  - Static versus dynamic priorities

- Utilization ($U$) and Schedulability
  - Main problem: Find $Bound$ for scheduling policy such that $U < Bound \Rightarrow$ All deadlines met!

- Optimality of EDF scheduling
  - $Bound_{EDF} = 100\%$
Schedulability Analysis of Periodic Tasks

Main problem:
- Given a set of periodic tasks, can they meet their deadlines?
- Depends on scheduling policy

Solution approaches
- Utilization bounds (Simplest)
- Exact analysis (NP-Hard)
- Heuristics

Two most important scheduling policies
- Earliest deadline first (Dynamic)
- Rate monotonic (Static)
Utilization Bounds

- Intuitively:
  - The lower the processor utilization, $U$, the easier it is to meet deadlines.
  - The higher the processor utilization, $U$, the more difficult it is to meet deadlines.
- Question: is there a threshold $U_{\text{bound}}$ such that
  - When $U < U_{\text{bound}}$ deadlines are met
  - When $U > U_{\text{bound}}$ deadlines are missed

Example
(Rate-Monotonic Scheduling)

Task 1
$p_1=2$
$c_1=1$

Task 2
$p_2=3$
$c_2=1.01$

$$U = \frac{c_1}{p_1} + \frac{c_2}{p_2} = \frac{1}{2} + \frac{1.01}{3} \approx 0.833$$

- Question: is there a threshold $U_{\text{bound}}$ such that
  - When $U < U_{\text{bound}}$ deadlines are met
  - When $U > U_{\text{bound}}$ deadlines are missed
Example
(Rate-Monotonic Scheduling)

Task 1
\( p_1 = 2 \)
\( c_1 = 1 \)

Task 2
\( p_2 = 3 \)
\( c_2 = 1.01 \)

\[ U = \frac{c_1}{p_1} + \frac{c_2}{p_2} = \frac{1}{2} + \frac{1.01}{3} \approx 83.3\% \]

Question: is there a threshold \( U_{\text{bound}} \) such that
- When \( U < U_{\text{bound}} \) deadlines are met
- When \( U > U_{\text{bound}} \) deadlines are missed

Example
(Rate-Monotonic Scheduling)

Task 1
\( p_1 = 2 \)
\( c_1 = 1 \)

Task 2
\( p_2 = 3 \)
\( c_2 = 1.01 \)

\[ U = \frac{c_1}{p_1} + \frac{c_2}{p_2} = \frac{1}{2} + \frac{1.01}{3} \approx 83.3\% \]

Question: is there a threshold \( U_{\text{bound}} \) such that
- When \( U < U_{\text{bound}} \) deadlines are met
- When \( U > U_{\text{bound}} \) deadlines are missed
Another Example
(Rate-Monotonic Scheduling)

Task 1
\[ P_1 = 2 \]
\[ C_1 = 1 \]

Task 2
\[ P_2 = 6 \]
\[ C_2 = 2.4 \]

\[ U = \frac{C_1}{P_1} + \frac{C_2}{P_2} = \frac{1}{2} + \frac{2.4}{6} = 90\% \]

Question: is there a threshold \( U_{\text{bound}} \) such that
- When \( U < U_{\text{bound}} \) deadlines are met
- When \( U > U_{\text{bound}} \) deadlines are missed

Another Example
(Rate-Monotonic Scheduling)

Task 1
\[ P_1 = 2 \]
\[ C_1 = 1 \]

Task 2
\[ P_2 = 6 \]
\[ C_2 = 2.4 \]

\[ U = \frac{C_1}{P_1} + \frac{C_2}{P_2} = \frac{1}{2} + \frac{2.4}{6} = 90\% \]

Question: is there a threshold \( U_{\text{bound}} \) such that
- When \( U < U_{\text{bound}} \) deadlines are met
- When \( U > U_{\text{bound}} \) deadlines are missed
Another Example
(Rate-Monotonic Scheduling)

Task 1
\[ P_1 = 2 \]
\[ C_1 = 1 \]

Task 2
\[ P_2 = 6 \]
\[ C_2 = 2.4 \]

\[ U = \frac{C_1}{P_1} + \frac{C_2}{P_2} = \frac{1}{2} + \frac{2.4}{6} = 0.9 \]

- Question: is there a threshold \( U_{\text{bound}} \) such that
  - When \( U < U_{\text{bound}} \) deadlines are met
  - When \( U > U_{\text{bound}} \) deadlines are missed

Schedulability depends on task set! No clean utilization threshold between schedulable and unschedulable task sets!
A Conceptual View of Schedulability

Utilization = \[ \sum_i \frac{C_i}{P_i} \]

Question: is there a threshold \( U_{\text{bound}} \) such that
- When \( U < U_{\text{bound}} \) deadlines are met
- When \( U > U_{\text{bound}} \) deadlines are missed

Modified Question: is there a threshold \( U_{\text{bound}} \) such that
- When \( U < U_{\text{bound}} \) deadlines are met
- When \( U > U_{\text{bound}} \) deadlines \text{may or may not} be missed
A Conceptual View of Schedulability

Utilization = \sum_{i} \frac{C_i}{P_i}

\text{Modified Question: is there a threshold } U_{\text{bound}} \text{ such that}

- When } U < U_{\text{bound}} \text{ deadlines are met}
- When } U > U_{\text{bound}} \text{ deadlines } \text{may or may not be} \text{ missed}

All green area (schedulable)
A Conceptual View of Schedulability

Equivalent question: What's the lowest utilization of an unschedulable task set?
(Called the Utilization Bound, $U_{\text{bound}}$)

All green area (schedulable)

Modified Question: is there a threshold $U_{\text{bound}}$ such that
- When $U < U_{\text{bound}}$ deadlines are met
- When $U > U_{\text{bound}}$ deadlines may or may not be missed

Solution Approach: Look at Critically-Schedulable Task Sets

Find some task set parameter $x$ such that
- Case (a): $x < x_o \Rightarrow U(x)$ decreases with $x$
- Case (b): $x > x_o \Rightarrow U(x)$ increases with $x$

Thus $U(x)$ is minimum when $x = x_o$
Find $U(x_o)$

Modified Question: is there a threshold $U_{\text{bound}}$ such that
- When $U < U_{\text{bound}}$ deadlines are met
- When $U > U_{\text{bound}}$ deadlines may or may not be missed
Deriving the Utilization Bound for Rate Monotonic Scheduling

Consider a simple case: 2 tasks

Find some task set parameter $x$ such that
Case (a): $x < x_o \Rightarrow U(x)$ decreases with $x$
Case (b): $x > x_o \Rightarrow U(x)$ increases with $x$
Thus $U(x)$ is minimum when $x = x_o$
Find $U(x_o)$
Consider a simple case: 2 tasks

Find some task set parameter $x$ such that

Case (a): $x < x_o \Rightarrow U(x)$ decreases with $x$

Case (b): $x > x_o \Rightarrow U(x)$ increases with $x$

Thus $U(x)$ is minimum when $x = x_o$

Find $U(x_o)$
Consider a simple case: 2 tasks

Find some task set parameter $x$ such that
Case (a): $x < x_o \implies U(x)$ decreases with $x$
Case (b): $x > x_o \implies U(x)$ increases with $x$
Thus $U(x)$ is minimum when $x = x_o$
Find $U(x_o)$
Deriving the Utilization Bound for Rate Monotonic Scheduling

Consider these two sub-cases:

**Case (a):**
\[ P_2 - \frac{P_2}{P_1} P_1 \]

**Task 1**
\[ P_1 \]

**Task 2**
\[ P_2 \]

**Case (b):**
\[ P_2 - \frac{P_2}{P_1} P_1 \]

**Task 1**
\[ P_1 \]

**Task 2**
\[ P_2 \]

Find some task set parameter \( x \) such that
- Case (a): \( x < x_o \) → \( U(x) \) decreases with \( x \)
- Case (b): \( x > x_o \) → \( U(x) \) increases with \( x \)
Thus \( U(x) \) is minimum when \( x = x_o \)
Find \( U(x_o) \)
Deriving the Utilization Bound for Rate Monotonic Scheduling

Consider these two sub-cases:

**Case (a):**
\[ C_1 \leq P_2 - \left\lfloor \frac{P_2}{P_1} \right\rfloor P_1 \]

**Case (b):**
\[ C_1 > P_2 - \left\lfloor \frac{P_2}{P_1} \right\rfloor P_1 \]

Find some task set parameter \( x \) such that:
- **Case (a):** \( x < x_o \) → \( U(x) \) decreases with \( x \)
- **Case (b):** \( x > x_o \) → \( U(x) \) increases with \( x \)

Thus \( U(x) \) is minimum when \( x = x_o \).

Find \( U(x_o) \).
Consider these two sub-cases:

**Case (a):**  \[ C_1 \leq P_2 - \left| \frac{P_2}{P_1} \right| P_1 \]

**Case (b):**  \[ C_1 > P_2 - \left| \frac{P_2}{P_1} \right| P_1 \]

Find some task set parameter \( x \) such that:
- For Case (a), \( x < x_0 \) implies \( U(x) \) decreases with \( x \).
- For Case (b), \( x > x_0 \) implies \( U(x) \) increases with \( x \).

Thus, \( U(x) \) is minimum when \( x = x_0 \).

Find \( U(x_0) \).
Deriving the Utilization Bound for Rate Monotonic Scheduling

Consider these two sub-cases:

**Case (a):**
\[ C_1 \leq P_2 - \left\lfloor \frac{P_2}{P_1} \right\rfloor P_1 \]

**Case (b):**
\[ C_1 > P_2 - \left\lfloor \frac{P_2}{P_1} \right\rfloor P_1 \]

The minimum utilization case:

\[ C_1 = P_2 - \left\lfloor \frac{P_2}{P_1} \right\rfloor P_1 \]

\[ U = 1 + \frac{C_1}{P_1} \left[ \frac{P_2}{P_1} - \left\lfloor \frac{P_2}{P_1} \right\rfloor - 1 \right] \]
Deriving the Utilization Bound for Rate Monotonic Scheduling

The minimum utilization case:

\[ C_i = P_i \left( \frac{P_i}{P_i} \right) \]

\[ C_i = P_i - \left( P_i \frac{P_i}{P_i} \right) \]

\[ U = 1 + \frac{P_i}{P_1} \left( \frac{P_2}{P_1} - \frac{P_2}{P_1} \right) \left( \frac{P_2}{P_1} - \frac{P_2}{P_1} \right) - 1 \]

\[ \Rightarrow \frac{P_2}{P_1} = 1 \]

\[ \Rightarrow U = 1 + \frac{P_i}{P_1} \left( \frac{P_2}{P_1} - 1 \right) \left( \frac{P_2}{P_1} - 2 \right) \]

\[ \frac{dU}{d\left( \frac{P_2}{P_1} \right)} = 0 \]

\[ \Rightarrow \frac{P_2}{P_1} = \frac{1}{2} \]

\[ \Rightarrow U = 0.83 \]

Note that \( C_i = P_2 - P_1 \)

Generalizing to N Tasks

\[ C_1 = P_2 - P_1 \]

\[ C_2 = P_3 - P_2 \]

\[ C_3 = P_4 - P_3 \]

\[ \ldots \]

\[ U = \frac{C_1}{P_1} + \frac{C_2}{P_2} + \frac{C_3}{P_3} + \ldots \]
Generalizing to N Tasks

\[
\begin{align*}
& C_1 = P_2 - P_1 \\
& C_2 = P_3 - P_2 \\
& C_3 = P_3 - P_2 \\
& \frac{dU}{d\left(\frac{P_2}{P_1}\right)} = 0 \\
& \frac{dU}{d\left(\frac{P_3}{P_2}\right)} = 0 \\
& \frac{dU}{d\left(\frac{P_3}{P_3}\right)} = 0 \\
& \Rightarrow \frac{P_{i+1}}{P_i} = 2^{\frac{1}{n}} \\
& \Rightarrow U = n \left( 2^{\frac{1}{n}} - 1 \right)
\end{align*}
\]
Generalizing to N Tasks

\[
C_1 = P_2 - P_1 \\
C_2 = P_3 - P_2 \\
C_3 = P_3 - P_2 \\
\ldots \\
C_n = P_n - P_{n-1} \\
\]

\[
\frac{dU}{d\left(\frac{P_i}{P_1}\right)} = 0 \\ \frac{dU}{d\left(\frac{P_i}{P_2}\right)} = 0 \quad \text{and} \quad \frac{dU}{d\left(\frac{P_i}{P_3}\right)} = 0 \quad \text{and} \quad \ldots \\
\Rightarrow \frac{P_{i+1}}{P_i} = 2^{\frac{1}{n}} \Rightarrow U = n\left(2^{\frac{1}{n}} - 1\right) \\
\quad \text{and} \quad \lim_{n \to \infty} \ U \to \ln 2
\]

Periodic Tasks

Periodic Task Scheduling

- Rate Monotonic
- EDF

- Bound
- Optimality
Periodic Tasks

Periodic Task Scheduling

Rate Monotonic

Bound

Optimality

EDF

Bound

Optimality

Coming Up

Periodic Task Scheduling

Rate Monotonic

Bound

Optimality

EDF

Bound

Optimality
Rate Monotonic Continued

- Rate monotonic scheduling is the optimal fixed-priority scheduling policy for periodic tasks.
  - Optimality (Trial #1):

  - Optimality (Trial #1): If any other fixed-priority scheduling policy can meet deadlines, so can RM.
Rate Monotonic Continued

- Rate monotonic scheduling is the optimal fixed-priority scheduling policy for periodic tasks.
  - Optimality (Trial #1): If any other fixed-priority scheduling policy can meet deadlines, so can RM.
Rate Monotonic Continued

- Rate monotonic scheduling is the optimal fixed-priority scheduling policy for periodic tasks.
  - Optimality (Trial #1): If any other fixed-priority scheduling policy can meet deadlines, so can RM.
Rate Monotonic Continued

- Rate monotonic scheduling is the optimal fixed-priority scheduling policy for periodic tasks.
  - Optimality (Trial #2): If any other fixed-priority scheduling policy can meet deadlines in the worst case scenario, so can RM.
- How to prove it?

Rate Monotonic Continued

- Rate monotonic scheduling is the optimal fixed-priority scheduling policy for periodic tasks.
  - Optimality (Trial #2): If any other fixed-priority scheduling policy can meet deadlines in the worst case scenario, so can RM.
- How to prove it?
  - Consider the worst case scenario
  - If someone else can schedule then RM can
The Worst-Case Scenario

Q: When does a periodic task, \( T \), experience the maximum delay?
A: When it arrives together with all the higher-priority tasks (critical instance)

Idea of Proof

If some higher-priority task does not arrive together with \( T \), aligning the arrival times can only increase the completion time of \( T \)

Proof (Case 1)

Case 1: higher priority task 1 is running when task 2 arrives
Case 1: higher priority task 1 is running when task 2 arrives
→ shifting task 1 right will increase completion time of 2
Proof (Case 2)

Task 1

Task 2

Case 2: processor is idle when task 2 arrives

Proof (Case 2)

Task 1

Task 2

Case 2: processor is idle when task 2 arrives

⇒ shifting task 1 left cannot decrease completion time of 2
Proof (Case 2)

Case 2: processor is idle when task 2 arrives
- shifting task 1 left cannot decrease completion time of 2

Optimality of Rate Monotonic
- If any other policy can meet deadlines so can RM

Policy X meets deadlines?
Optimality of Rate Monotonic

- If any other policy can meet deadlines so can RM

   Policy X meets deadlines? **YES**
   → RM meets deadlines

Coming Up

Periodic Task Scheduling

- Rate Monotonic
  - Bound
  - Optimality
- EDF
  - Bound
  - Optimality
Coming Up

Periodic Task Scheduling

Rate Monotonic

EDF

Bound
Optimality
Bound
Optimality

Utilization Bound of EDF

- Why is it 100%?
- Consider a task set where:

\[ \sum_{i} \frac{C_i}{P_i} = 1 \]

- Imagine a policy that reserves for each task \( i \) a fraction \( f_i \) of each clock tick, where \( f_i = \frac{C_i}{P_i} \)
Imagine a policy that reserves for each task $i$ a fraction $f_i$ of each time unit, where $f_i = C_i / P_i$.

This policy meets all deadlines, because within each period $P$, it reserves for task $i$ a total time 

$$
\text{Time} = f_i P_i = \left( \frac{C_i}{P_i} \right) P_i = C_i \quad \text{(i.e., enough to finish)}
$$
Utilization Bound of EDF

- Pick any two execution chunks that are not in EDF order and swap them

- Still meets deadlines!

Utilization Bound of EDF

- Pick any two execution chunks that are not in EDF order and swap them

- Still meets deadlines!
- Repeat swap until all in EDF order
  \( \rightarrow \) EDF meets deadlines
Periodic Tasks

Periodic Task Scheduling

Rate Monotonic
Bound Optimality

EDF
Bound Optimality

Done

Periodic Task Scheduling

Rate Monotonic
Bound Optimality

EDF
Bound Optimality
Exercise:
Know Your Worst Case Scenario

- Consider a periodic system of two tasks
- Let \( U_i = C_i / P_i \) (for \( i = 1,2 \))
- What is the maximum value of:
  \[ \Pi (1 + U_i) \]
  for a schedulable system?

Deriving the Utilization Bound
for Rate Monotonic Scheduling

- The minimum utilization case:
  \[ c_i = P_i \left( \frac{P_i}{C_i} \right) \]
  \[ C_i = P_i - \left( \frac{P_i}{P_i} \right) \]
  \[ U = 1 + \frac{P_1}{P_2} \left( \frac{P_2}{P_1} \right) - \left( \frac{P_2}{P_1} \right) \left( \frac{P_2}{P_1} - 1 \right) \]
  \[ \Rightarrow \left( \frac{P_2}{P_1} \right) = 1 \]
Deriving the Utilization Bound for Rate Monotonic Scheduling

The minimum utilization case:

\[ U = 1 + C_1 \left( \frac{P_2}{P_1} \right) - \left( \frac{P_2}{P_1} \right) - 1 \]

\[ C_1 = P_2 - \left( \frac{P_2}{P_1} \right) P_1 \]

\[ U = 1 + P_1 \left( \frac{P_2}{P_1} \right) - \left( \frac{P_2}{P_1} \right) - \left( \frac{P_2}{P_1} \right) = 1 \]

\[ \Rightarrow \left( \frac{P_2}{P_1} \right) = 1 \]

\[ Task 1 \]
\[ Task 2 \]
Solutions

\[ \begin{align*}
C_1 &= P_2 - P_1 \\
C_2 &= P_1 - C_1 = 2P_1 - P_2
\end{align*} \]

Critically Schedulable

Schedulable
Solutions

Critically Schedulable

\[
\begin{align*}
C_1 &= P_2 - P_1 \\
C_2 &= P_1 - C_1 = 2P_1 - P_2 \\
U_1 + 1 &= \frac{C_1}{P_1} + 1 = \frac{C_1 + P_1}{P_1} = \frac{P_2}{P_1} \\
U_2 + 1 &= \frac{C_2}{P_2} + 1 = \frac{C_2 + P_2}{P_2} = \frac{2P_1}{P_2}
\end{align*}
\]

Schedulable

\[\prod_i (U_i + 1) = 2\]

\[\prod_i (U_i + 1) \leq 2\]
The General Case

\[
C_i = P_{i+1} - P_i \\
C_n = 2P_1 - P_n
\]

Critically Schedulable

Schedulable

The General Case

\[
C_i = P_{i+1} - P_i \\
C_n = 2P_1 - P_n \\
U_i + 1 = \frac{C_i}{P_i} + 1 = \frac{C_i + P_i}{P_i} = \frac{P_{i+1}}{P_i}
\]

Critically Schedulable

Schedulable
The General Case

\[ C_i = P_{i+1} - P_i \]
\[ C_n = 2P_1 - P_n \]
\[ U_i + 1 = \frac{C_i}{P_i} + 1 = \frac{C_i + P_i}{P_i} = \frac{P_{i+1}}{P_i} \]
\[ U_n + 1 = \frac{C_n}{P_n} + 1 = \frac{C_n + P_n}{P_n} = \frac{2P_1}{P_n} \]

Critically Schedulable

Schedulable

The General Case

\[ C_i = P_{i+1} - P_i \]
\[ C_n = 2P_1 - P_n \]
\[ U_i + 1 = \frac{C_i}{P_i} + 1 = \frac{C_i + P_i}{P_i} = \frac{P_{i+1}}{P_i} \]
\[ U_n + 1 = \frac{C_n}{P_n} + 1 = \frac{C_n + P_n}{P_n} = \frac{2P_1}{P_n} \]
\[ \prod_i(U_i + 1) = \frac{P_2}{P_1} \frac{P_3}{P_2} ... \frac{P_n}{P_{n-1}} \frac{2P_1}{P_n} = 2 \]
\[ \prod_i(U_i + 1) \leq 2 \]
A set of periodic tasks is schedulable if:

\[ \prod_i (U_i + 1) \leq 2 \]

It's a better bound than \[ \sum_i U_i \leq n \left( 2^{1/n} - 1 \right) \]

Example:
- A system of two tasks with \( U_1 = 0.8, \ U_2 = 0.1 \)
The Hyperbolic Bound for Rate Monotonic Scheduling

- A set of periodic tasks is schedulable if:
  \[ \prod (U_i + 1) \leq 2 \]

- It’s a better bound!
  - Example:
    - A system of two tasks with \( U_1 = 0.8, \ U_2 = 0.1 \)
    - Liu and Layland bound: \( U_1 + U_2 = 0.9 > 0.83 \)
    - Hyperbolic bound \((U_1+1)(U_2+1) = 1.8 \times 1.1 = 1.98 < 2\)
Scheduling Taxonomy

Periodic Task Scheduling

- Rate Monotonic
  - Bound
  - Optimality
- EDF
  - Bound
  - Optimality

Hyperbolic Bound