Resource Augmentation

Prof. Dr. Jian-Jia Chen

LS 12, TU Dortmund

28 April 2015

Schedulability Condition for Rate Monotonic

The time-demand function $W_i(t)$ of the task τ_i is defined as follows:

$$W_i(t) = C_i + \sum_{j=1}^{i-1} \left\lceil \frac{t}{T_j} \right\rceil C_j.$$

Theorem

A system $\mathbf{T}$ of periodic, independent, preemptable tasks is schedulable on one processor by rate monotonic scheduling if

$$\forall \tau_i \in \mathbf{T} \; \exists t \text{ with } 0 < t \leq D_i \text{ and } W_i(t) \leq t.$$

Schedulability Condition for Rate Monotonic

The time-demand function $W_i(t)$ of the task τ_i is defined as follows:

$$W_i(t) = C_i + \sum_{j=1}^{i-1} \left\lceil \frac{t}{T_j} \right\rceil C_j.$$

Theorem

A system $\mathbf{T}$ of periodic, independent, preemptable tasks is schedulable on one processor by rate monotonic scheduling if

$$\forall \tau_i \in \mathbf{T} \; \exists t \text{ with } 0 < t \leq D_i \text{ and } W_i(t) \leq t.$$

Theorem

Eisenbrand and Rothvoss [RTSS 2008]: Fixed-Priority Real-Time Scheduling: Response Time Computation Is $\mathcal{N}P$-Hard

Schedulability Conditions for EDF Scheduling

Theorem

A task set **T** of independent, preemptable, periodic tasks with relative deadlines equal to or less than their periods can be feasibly scheduled (under EDF) on one processor if and only if

$$\forall t \geq 0, \sum_{i=1}^{n} dbf(\tau_i, t) = \sum_{i=1}^{n} \left\lfloor \frac{t + T_i - D_i}{T_i} \right\rfloor C_i \leq t,$$

where $dbf(\tau_i, t) = \left\lfloor \frac{t+T_i-D_i}{T_i} \right\rfloor C_i$ is the definition of the demand bound function of task τ_i at time t.

Schedulability Conditions for EDF Scheduling

Theorem

A task set $\mathbf{T}$ of independent, preemptable, periodic tasks with relative deadlines equal to or less than their periods can be feasibly scheduled (under EDF) on one processor if and only if

$$\forall t \geq 0, \sum_{i=1}^{n} dbf(\tau_i, t) = \sum_{i=1}^{n} \left\lfloor \frac{t + T_i - D_i}{T_i} \right\rfloor C_i \leq t,$$

where $dbf(\tau_i, t) = \left\lfloor \frac{t+T_i-D_i}{T_i} \right\rfloor C_i$ is the definition of the demand bound function of task τ_i at time t.

Theorem

Eisenbrand and Rothvoss [SODA 2010]: testing EDF schedulability of such a task set is (weakly) co$\mathcal{NP}$-hard. That is, deciding whether a task set is not schedulable by EDF is (weakly) $\mathcal{NP}$-hard.

Schedulability

- The issue for uniprocessor scheduling is how to analyze the schedulability.
  - EDF is optimal
  - DM is optimal for fixed-priority scheduling when $D_i \leq T_i$
  - Ausley's iterative approach (1992) can also be applied for fixed-priority scheduling when $D_i > T_i$
- As verifying the schedulability is $\mathcal{N}P$-hard or co$\mathcal{N}P$-hard, there does not exist any polynomial-time algorithm for schedulability tests unless $\mathcal{P} = \mathcal{N}P$.

Schedulability

- The issue for uniprocessor scheduling is how to analyze the schedulability.
  - EDF is optimal
  - DM is optimal for fixed-priority scheduling when $D_i \leq T_i$
  - Ausley's iterative approach (1992) can also be applied for fixed-priority scheduling when $D_i > T_i$
- As verifying the schedulability is $\mathcal{N}P$-hard or co$\mathcal{N}P$-hard, there does not exist any polynomial-time algorithm for schedulability tests unless $\mathcal{P} = \mathcal{N}P$.
- Approximations are possible, but what do we approximate when only binary decisions (Yes or No) have to be made?
  - Answers like probabilistic guarantee are unlikely preferred, e.g., the task set is 99% schedulable.
  - Deadline relaxation: requires modifications of task specification
  - Period relaxation: requires modifications of task specification
  - Resource augmentation by speeding up: a faster platform
  - Resource augmentation by allocating more processors: a better platform

Schedulability

- The issue for uniprocessor scheduling is how to analyze the schedulability.
  - EDF is optimal
  - DM is optimal for fixed-priority scheduling when $D_i \leq T_i$
  - Ausley's iterative approach (1992) can also be applied for fixed-priority scheduling when $D_i > T_i$
- As verifying the schedulability is $\mathcal{N}P$-hard or co$\mathcal{N}P$-hard, there does not exist any polynomial-time algorithm for schedulability tests unless $\mathcal{P} = \mathcal{N}P$.
- Approximations are possible, but what do we approximate when only binary decisions (Yes or No) have to be made?
  - Answers like probabilistic guarantee are unlikely preferred, e.g., the task set is 99% schedulable.
  - Deadline relaxation: requires modifications of task specification
  - Period relaxation: requires modifications of task specification
  - Resource augmentation by speeding up: a faster platform
  - Resource augmentation by allocating more processors: a better

Resource Augmentation

For an algorithm A with a resource augmentation factor ρ, it guarantees that

$\Longrightarrow$

if the task set (system) is schedulable (feasible), Algorithm A will also returns a schedulable (feasible) answer when speeding up the system by a factor ρ, or

Resource Augmentation

For an algorithm A with a resource augmentation factor ρ, it guarantees that

$\Rightarrow$

if the task set (system) is schedulable (feasible), Algorithm A will also returns a schedulable (feasible) answer when speeding up the system by a factor ρ, or

$\Leftarrow$

if Algorithm A does not return a schedulable (feasible) answer, the system is also unschedulable (infeasible) when slowing down by a factor ρ.

Schedulability by Least Utilization Bound

Algorithm: Given n periodic tasks with relative deadline equal to the period

- If the total utilization is less than $n(2^{\frac{1}{n}} - 1)$, the task set is schedulable;
- otherwise, the task set is probably not schedulable.

Schedulability by Least Utilization Bound

Algorithm: Given n periodic tasks with relative deadline equal to the period

- If the total utilization is less than $n(2^{\frac{1}{n}} - 1)$, the task set is schedulable;
- otherwise, the task set is probably not schedulable.

The algorithm has resource augmentation factor $\frac{1}{0.693}$ (or $\frac{1}{\ln 2}$) to decide whether a task set is schedulable by the rate monotonic scheduling algorithm.

Schedulability by Least Utilization Bound

Algorithm: Given n periodic tasks with relative deadline equal to the period

- If the total utilization is less than $n(2^{\frac{1}{n}} - 1)$, the task set is schedulable;
- otherwise, the task set is probably not schedulable.

The algorithm has resource augmentation factor $\frac{1}{0.693}$ (or $\frac{1}{\ln 2}$) to decide whether a task set is schedulable by the rate monotonic scheduling algorithm.

- The resource augmentation factor analysis is tight
  - n jobs with the same parameters $C = (2^{\frac{1}{n}} - 1) + \epsilon, D = P = 1$ where $\epsilon > 0$ and $\epsilon \rightarrow^{+} 0$.
  - The task set is schedulable, but the above testing algorithm says that it is probably not schedulable.

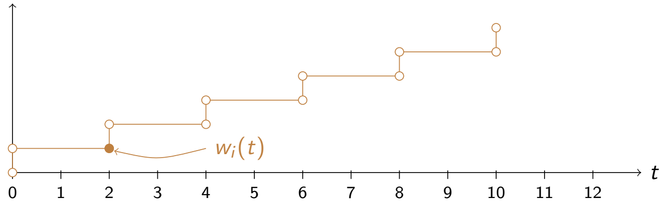
Time Demand Function Revisit for RM/DM

Let $w_i(t)$ of the task τ_i be defined as follows:

$$w_i(t) = \left\lceil \frac{t}{T_i} \right\rceil C_i.$$

We need approximation to enforce a *polynomial-time* schedulability test.

$$w_i^*(t) = C_i + \frac{t}{T_i} C_i.$$



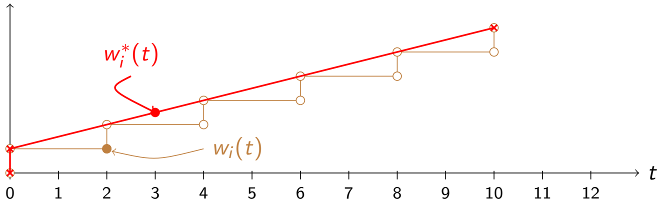
Time Demand Function Revisit for RM/DM

Let $w_i(t)$ of the task τ_i be defined as follows:

$$w_i(t) = \left\lceil \frac{t}{T_i} \right\rceil C_i.$$

We need approximation to enforce a *polynomial-time* schedulability test.

$$w_i^*(t) = C_i + \frac{t}{T_i} C_i.$$



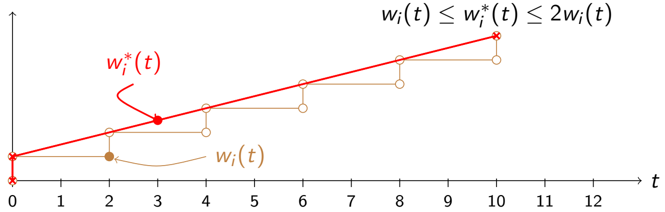
Time Demand Function Revisit for RM/DM

Let $w_i(t)$ of the task τ_i be defined as follows:

$$w_i(t) = \left\lceil \frac{t}{T_i} \right\rceil C_i.$$

We need approximation to enforce a *polynomial-time* schedulability test.

$$w_i^*(t) = C_i + \frac{t}{T_i} C_i.$$



Resource Augmentation

The approximated time-demand function $W_i^*(t)$ of τ_i is defined as follows:

$$W_i^*(t) = C_i + \sum_{j=1}^{i-1} w_j^*(t).$$

- If $W_i^*(t) \leq t$, then $W_i(t) \leq t$.
- If $W_i^*(t) > t$, then $W_i(t) > 0.5t$.

Theorem

[Fisher and Baruah, 2005] A system $\mathbf{T}$ of periodic, independent, preemptable tasks is schedulable on one processor by RM/DM if

$$\forall \tau_i \in \mathbf{T}\ \exists t \text{ with } 0 < t \leq D_i \text{ and } W_i^*(t) \leq t.$$

Otherwise, the system is not schedulable when slowing down by a factor 2 (i.e.,running at 0.5 of the original speed).

Resource Augmentation

The approximated time-demand function $W_i^*(t)$ of τ_i is defined as follows:

$$W_i^*(t) = C_i + \sum_{j=1}^{i-1} w_j^*(t).$$

- If $W_i^*(t) \leq t$, then $W_i(t) \leq t$.
- If $W_i^*(t) > t$, then $W_i(t) > 0.5t$.

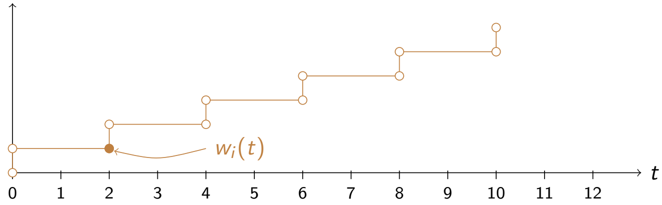
The analysis is tight by considering the following example:

- A task with period $P = D = 1$ and $C = 0.5 + \epsilon$.
- Since $(0.5 + \epsilon)(1 + x) > x$ for all $x \geq 0$ and $\epsilon > 0$, the above test does not succeed.
- The system is still schedulable if it is slowed down to run at $0.5 + \epsilon$ of the original speed.

Time Demand Function Revisit for RM/DM

Given a precision factor δ, we can approximation $\left\lceil \frac{t}{T_j} \right\rceil$ by $w'_j(t)$

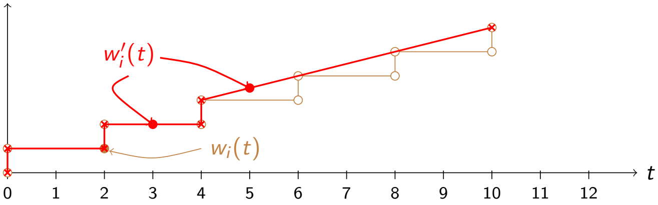
$$w'_j(t) = \begin{cases} \left\lceil \frac{t}{T_j} \right\rceil C_j & \text{if } t \leq (\lceil \frac{1}{\delta} \rceil - 2) T_j \\ (1 + \frac{t}{T_j}) C_j & \text{if } t > (\lceil \frac{1}{\delta} \rceil - 2) T_j \end{cases}$$



Time Demand Function Revisit for RM/DM

Given a precision factor δ, we can approximation $\left\lceil \frac{t}{T_j} \right\rceil$ by $w'_j(t)$

$$w'_j(t) = \begin{cases} \left\lceil \frac{t}{T_j} \right\rceil C_j & \text{if } t \leq (\lceil \frac{1}{\delta} \rceil - 2) T_j \\ (1 + \frac{t}{T_j}) C_j & \text{if } t > (\lceil \frac{1}{\delta} \rceil - 2) T_j \end{cases}$$



Resource Augmentation

The approximated time-demand function $W_i'(t)$ of τ_i is defined as follows:

$$W_i'(t) = C_i + \sum_{j=1}^{i-1} w_j'(t).$$

- If $W_i'(t) \leq t$, then $W_i(t) \leq t$.
- If $W_i'(t) > t$, then $W_i(t) > (1 - \frac{1}{\lceil \frac{1}{\delta} \rceil})t$.

Theorem

[Fisher and Baruah, 2005]A system $\mathbf{T}$ of periodic, independent, pre-emptable tasks is schedulable on one processor by RM/DM if

$$\forall \tau_i \in \mathbf{T}\ \exists t \text{ with } 0 < t \leq D_i \text{ and } W_i'(t) \leq t.$$

Otherwise, the system is not schedulable when slowing down to run at speed $(1 - \delta)$ of the original speed.

Exercise

Exercise

Please provide pseudo code and analyze the complexity and resource augmentation factor so that

- the algorithm runs in polynomial time with respect to $\frac{1}{\delta}$ and the number of tasks, and
- the resource augmentation factor is $\frac{1}{1-\delta}$.

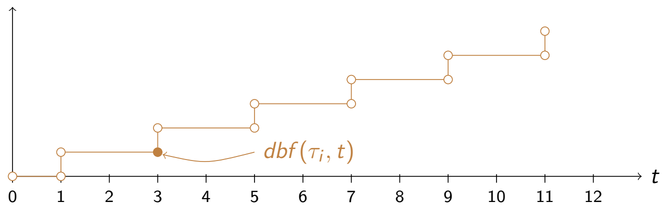
Demand Bound Function Revisit for EDF

Define demand bound function $dbf(\tau_i, t)$ as

$$dbf(\tau_i, t) = \max\left\{0, \left\lfloor \frac{t + T_i - D_i}{T_i} \right\rfloor\right\} C_i = \max\left\{0, \left\lfloor \frac{t - D_i}{T_i} \right\rfloor + 1\right\} C_i.$$

We need approximation to enforce a *polynomial-time* schedulability test.

$$dbf^*(\tau_i, t) = \begin{cases} 0 & \text{if } t < D_i \\ (\frac{t-D_i}{T_i} + 1)C_i & \text{otherwise.} \end{cases}$$



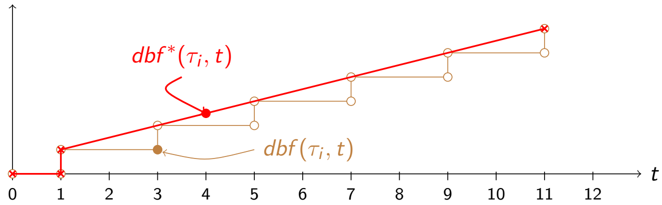
Demand Bound Function Revisit for EDF

Define demand bound function $dbf(\tau_i, t)$ as

$$dbf(\tau_i, t) = \max\left\{0, \left\lfloor \frac{t + T_i - D_i}{T_i} \right\rfloor\right\} C_i = \max\left\{0, \left\lfloor \frac{t - D_i}{T_i} \right\rfloor + 1\right\} C_i.$$

We need approximation to enforce a *polynomial-time* schedulability test.

$$dbf^*(\tau_i, t) = \begin{cases} 0 & \text{if } t < D_i \\ (\frac{t-D_i}{T_i} + 1)C_i & \text{otherwise.} \end{cases}$$



Demand Bound Function Revisit for EDF

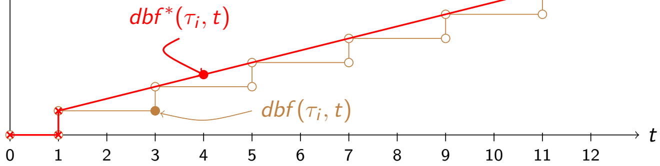
Define demand bound function $dbf(\tau_i, t)$ as

$$dbf(\tau_i, t) = \max\left\{0, \left\lfloor \frac{t + T_i - D_i}{T_i} \right\rfloor\right\} C_i = \max\left\{0, \left\lfloor \frac{t - D_i}{T_i} \right\rfloor + 1\right\} C_i.$$

We need approximation to enforce a *polynomial-time* schedulability test.

$$dbf^*(\tau_i, t) = \begin{cases} 0 & \text{if } t < D_i \\ (\frac{t-D_i}{T_i} + 1)C_i & \text{otherwise.} \end{cases}$$

$$dbf(\tau_i, t) \leq dbf^*(\tau_i, t) \leq 2dbf(\tau_i, t)$$



Resource Augmentation by EDF

- If $\sum_{i=1}^{n} dbf^*(\tau_i, t) \leq t$, then $\sum_{i=1}^{n} dbf(\tau_i, t) \leq t$.
- If $\sum_{i=1}^{n} dbf^*(\tau_i, t) > t$, then $\sum_{i=1}^{n} dbf(\tau_i, t) > 0.5t$.

With similar strategy, we can prove that such an approach has a resource augmentation factor 2.

- For all t, if $\sum_{i=1}^{n} dbf^*(\tau_i, t) \leq t$, then it is schedulable by EDF;
- otherwise, it is probably not schedulable.

Resource Augmentation by EDF

- If $\sum_{i=1}^{n} dbf^*(\tau_i, t) \leq t$, then $\sum_{i=1}^{n} dbf(\tau_i, t) \leq t$.
- If $\sum_{i=1}^{n} dbf^*(\tau_i, t) > t$, then $\sum_{i=1}^{n} dbf(\tau_i, t) > 0.5t$.

With similar strategy, we can prove that such an approach has a resource augmentation factor 2.

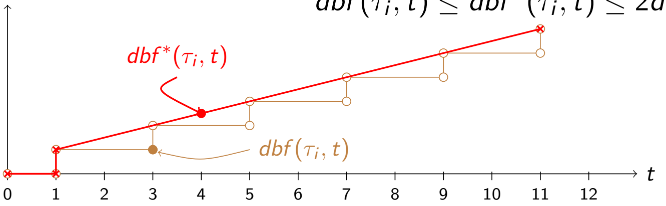
- For all t, if $\sum_{i=1}^{n} dbf^*(\tau_i, t) \leq t$, then it is schedulable by EDF;
- otherwise, it is probably not schedulable.

Similarly, we can also extend to approximate with a given error tolerate parameter δ. [Albers and Slomka, 2004]

Is the Approximation for EDF Tight?

$$dbf^*(\tau_i, t) = \begin{cases} 0 & \text{if } t < D_i \\ (\frac{t-D_i}{T_i} + 1)C_i & \text{otherwise.} \end{cases}$$

$$dbf(\tau_i, t) \leq dbf^*(\tau_i, t) \leq 2dbf(\tau_i, t)$$



- Not really, when t is very close to $t + D_i$, we can find a sharp increase of the demand bound function.
- Even though a factor 2 in is tight to bound dbf and dbf^*, it is not tight for resource augmentation even for a uniprocessor system.

Resource Augmentation for EDF

Theorem

Chen and Chakraborty [RTSS 2011]

- There exists a set of input instances such that the resource augmentation factor for one-step approximation of DBF is 1.5.
- There resource augmentation factor for one-step approximation of DBF is at most $\frac{2e-1}{e} \approx 1.6322$.

Proofs and details are omitted.