k^2U and k^2Q: A General Framework from k-Point Effective Schedulability Analysis to Utilization-Based Tests

Prof. Dr. Jian-Jia Chen

TU Dortmund

30/31,05,2016

Outline

Outline

Introduction

Utilization-Based Analytical Framework

Selected Applications

Conclusions

Schedulability Test of Fixed-Priority (FP) Scheduling

Time Demand Analysis (TDA): Task τ_k (with $D_i = T_i$) can be feasibly scheduled by a fixed-priority scheduling algorithm if

$$\exists t \text{ with } 0 < t \leq T_k \;\; \textit{and} \;\; C_k + \sum_{j=1}^{k-1} \left\lceil \frac{t}{T_j} \right\rceil C_j \leq t.$$

(I will implicitly assume $k-1$ higher-priority tasks.)

Schedulability Test of Fixed-Priority (FP) Scheduling

Time Demand Analysis (TDA): Task τ_k (with $D_i = T_i$) can be feasibly scheduled by a fixed-priority scheduling algorithm if

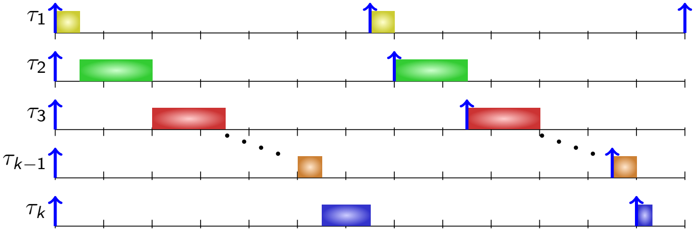
$$\exists t \text{ with } 0 < t \leq T_k \textit{ and } C_k + \sum_{j=1}^{k-1} \left\lceil \frac{t}{T_j} \right\rceil C_j \leq t.$$

(I will implicitly assume $k-1$ higher-priority tasks.)

- This test takes pseudo-polynomial time
- If C_k is small enough, it always answers "schedulable".
- Can we derive such a bound of C_k *efficiently*?

(Liu and Layland-Bound) Structure

The non-schedulability of task τ_k in rate-monotonic scheduling (RM) implies the following structure if $2T_1 \geq T_k$:



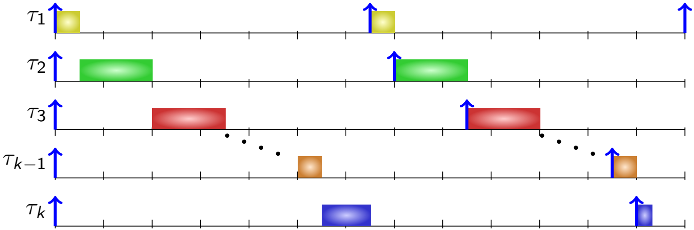
Minimize $\sum_{i=1}^{k} U_i$

$$C_k + \sum_{i=1}^{k-1} C_i + \sum_{i=0}^{j-1} C_i > T_j, \forall j = 1, 2, \ldots, k-1,$$

$$C_k + 2\sum_{i=1}^{k-1} C_i > T_k.$$

(Liu and Layland-Bound) Structure

The non-schedulability of task τ_k in rate-monotonic scheduling (RM) implies the following structure if $2T_1 \geq T_k$:



Minimize $\sum_{i=1}^{k} U_i$

$$C_k + \sum_{i=1}^{k-1} T_i U_i + \sum_{i=0}^{j-1} T_i U_i > T_j, \forall j = 1, 2, \ldots, k-1,$$

$$C_k + 2\sum_{i=1}^{k-1} T_i U_i > T_k.$$

Key Questions

Does there exist a unified utilization-based analysis, regardless of the platform model or the task model?

Key Questions

Does there exist a unified utilization-based analysis, regardless of the platform model or the task model?

How can we find the bound of C_k *efficiently*?

Can we build utilization-based analysis *almost automatically* by referring to a schedulability test?

Outline

Introduction

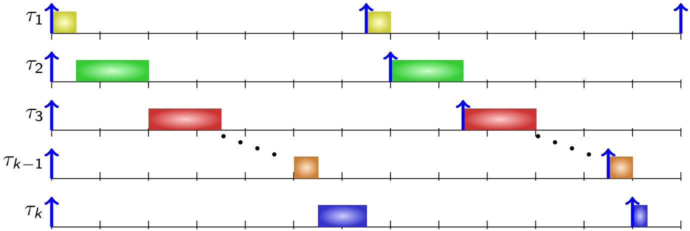
Utilization-Based Analytical Framework

Selected Applications

Conclusions

Revisit (Liu and Layland-Bound) Structure

- $C_i = T_i U_i$
- The non-schedulability of τ_k implies such a structure if $2T_1 \geq T_k$:



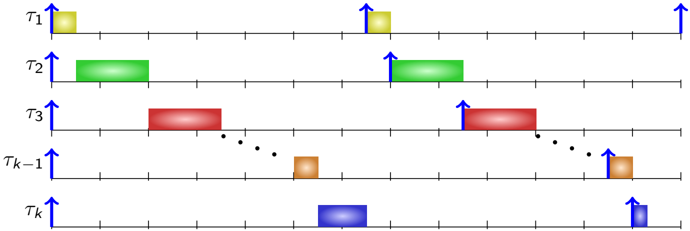
Objective is to find the minimum C_k such that

$$C_k + \sum_{i=1}^{k-1} C_i + \sum_{i=0}^{j-1} C_i > T_j, \forall j = 1, 2, \ldots, k-1,$$

$$C_k + 2\sum_{i=1}^{k-1} C_i > T_k.$$

Revisit (Liu and Layland-Bound) Structure

- $C_i = T_i U_i$
- The non-schedulability of τ_k implies such a structure if $2T_1 \geq T_k$:



Objective is to find the minimum C_k such that

$$C_k + \sum_{i=1}^{k-1} T_i U_i + \sum_{i=0}^{j-1} T_i U_i > T_j, \forall j = 1, 2, \ldots, k-1,$$

$$C_k + 2 \sum_{i=1}^{k-1} T_i U_i > T_k.$$

k-Point Effective Schedulability Test

Let us replace $T_i > 0$ with $t_i > 0$ (as a variable)

Definition

A k-point effective schedulability test is

- a sufficient test by verifying the existence of $t_j \in \{t_1, t_2, \dots t_k\}$ with $0 \leq t_1 \leq t_2 \leq \cdots \leq t_{k-1} \leq t_k$
- such that

$$C_k + \sum_{i=1}^{k-1} t_i U_i + \sum_{i=1}^{j-1} t_i U_i \leq t_j.$$

k-Point Effective Schedulability Test

Let us replace $T_i > 0$ with $t_i > 0$ (as a variable)

Definition

A k-point effective schedulability test is

- a sufficient test by verifying the existence of $t_j \in \{t_1, t_2, \ldots t_k\}$ with $0 \leq t_1 \leq t_2 \leq \cdots \leq t_{k-1} \leq t_k$
- such that

$$C_k + \sum_{i=1}^{k-1} \alpha_i t_i U_i + \sum_{i=1}^{j-1} \beta_i t_i U_i \leq t_j.$$

In the above Liu&Layland task model as an example:

- $\alpha_i = 1, \beta_i = 1, \forall i = 1, 2, \ldots, k-1$
- $t_i = T_i, \forall i = 1, 2, \ldots, k$

Key Result

Suppose a given k-point effective schedulability test of a scheduling algorithm, in which $0 < \alpha_i \leq \alpha$, and $0 < \beta_i \leq \beta$ for any $i = 1, 2, \ldots, k-1$, $0 < t_k$.

Key Result

Suppose a given k-point effective schedulability test of a scheduling algorithm, in which $0 < \alpha_i \leq \alpha$, and $0 < \beta_i \leq \beta$ for any $i = 1, 2, \ldots, k-1$, $0 < t_k$.

Lemma

Lemma 1 Task τ_k is schedulable by the scheduling algorithm if the following condition holds

$$\frac{C_k}{t_k} \leq \frac{\frac{\alpha}{\beta} + 1}{\prod_{j=1}^{k-1} (\beta U_j + 1)} - \frac{\alpha}{\beta}.$$

Key Result

Suppose a given k-point effective schedulability test of a scheduling algorithm, in which $0 < \alpha_i \leq \alpha$, and $0 < \beta_i \leq \beta$ for any $i = 1, 2, \ldots, k-1$, $0 < t_k$.

Lemma

Lemma 1 Task τ_k is schedulable by the scheduling algorithm if the following condition holds

$$\left(\frac{C_k}{t_k} + \frac{\alpha}{\beta}\right) \prod_{j=1}^{k-1} (\beta U_j + 1) \leq \frac{\alpha}{\beta} + 1$$

Key Result

Suppose a given k-point effective schedulability test of a scheduling algorithm, in which $0 < \alpha_i \leq \alpha$, and $0 < \beta_i \leq \beta$ for any $i = 1, 2, \ldots, k-1$, $0 < t_k$.

Lemma

Lemma 1 Task τ_k is schedulable by the scheduling algorithm if the following condition holds

$$\left(\frac{C_k}{t_k} + \frac{\alpha}{\beta}\right) \prod_{j=1}^{k-1} (\beta U_j + 1) \leq \frac{\alpha}{\beta} + 1$$

In the above Liu&Layland task model as an example:

- $\alpha = 1, \beta = 1$
- $t_k = T_k$
- Hyperbolic Bound: $\prod_{j=1}^{k} (U_j + 1) \leq 2$.

A Sketched Proof

The unschedulability implies that $C_k > C_k^*$, where C_k^* is defined in the following optimization problem:

$$
\begin{aligned}
\text{infimum} \quad & C_k^* \\
\text{such that} \quad & C_k^* + \sum_{i=1}^{k-1} \alpha t_i^* U_i + \sum_{i=1}^{j-1} \beta t_i^* U_i > t_j^*, && \forall j = 1, 2, \ldots, k \\
& t_j^* \geq 0, && \forall j = 1, 2, \ldots, k,
\end{aligned}
$$

where $t_1^*, t_2^*, \ldots, t_{k-1}^*$ and C_k^* are variables, α, β are constants, and t_k^* is defined as t_k.

A Sketched Proof

The unschedulability implies that $C_k > C_k^*$, where C_k^* is defined in the following optimization problem:

$$
\begin{aligned}
\text{infimum} \quad & C_k^* \\
\text{such that} \quad & C_k^* + \sum_{i=1}^{k-1} \alpha t_i^* U_i + \sum_{i=1}^{j-1} \beta t_i^* U_i > t_j^*, & \forall j = 1, 2, \ldots, k \\
& t_j^* \geq 0, & \forall j = 1, 2, \ldots, k,
\end{aligned}
$$

where $t_1^*, t_2^*, \ldots, t_{k-1}^*$ and C_k^* are variables, α, β are constants, and t_k^* is defined as t_k.

The above linear programming gives the minimum C_k^* to be unschedulable. Therefore, if $C_k \leq C_k^*$, task τ_k is guaranteed to be schedulable.

Framework (First Part)

Demonstrated Applications:

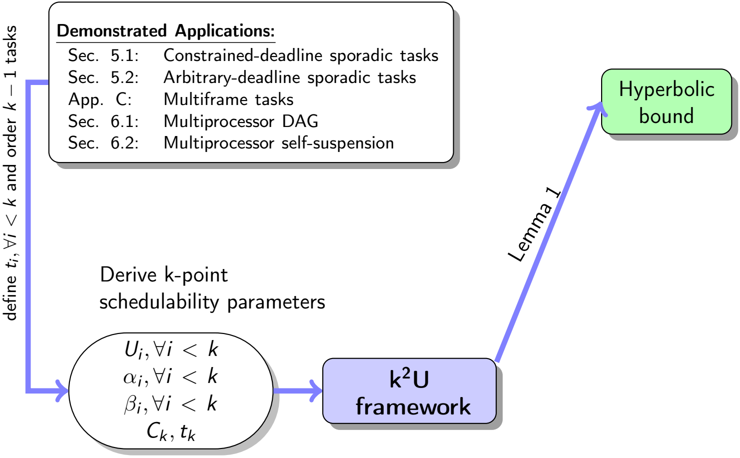
Sec. 5.1:	Constrained-deadline sporadic tasks
Sec. 5.2:	Arbitrary-deadline sporadic tasks
App. C:	Multiframe tasks
Sec. 6.1:	Multiprocessor DAG
Sec. 6.2:	Multiprocessor self-suspension

define $t_i, \forall i < k$ and order $k-1$ tasks

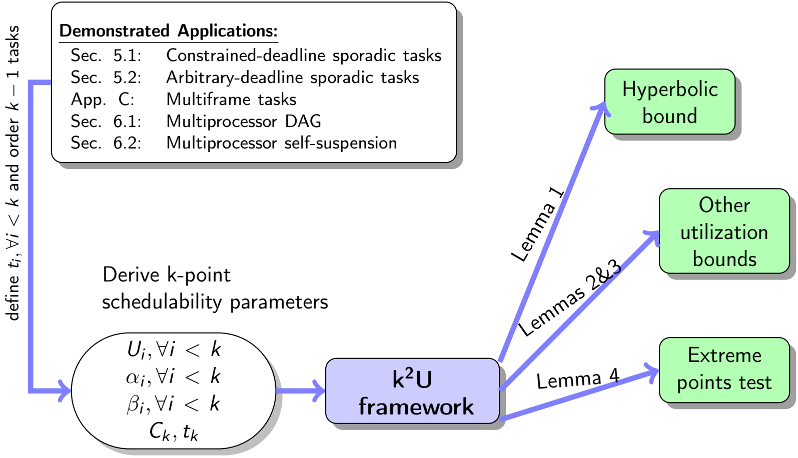
Derive k-point
schedulability parameters

$U_i, \forall i < k$
$\alpha_i, \forall i < k$
$\beta_i, \forall i < k$
C_k, t_k

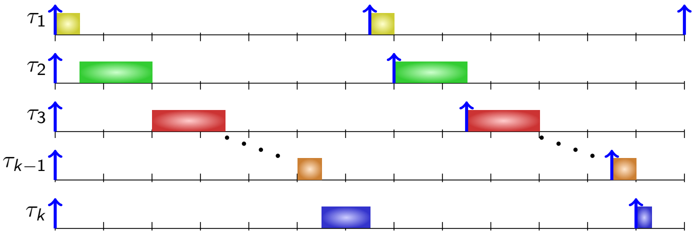
Framework (First Part)



Framework (First Part)



Hyperbolic Bound for Sporadic Task Systems



Let t_i be $\left\lfloor \frac{T_k}{T_i} \right\rfloor T_i$. Therefore, we have $\alpha_i = 1$ and $\beta_i \leq 1$.

Theorem

[Bini and Buttazzo, ECRTS 2001] Task τ_k is schedulable by RM on a uniprocessor system if

$$\prod_{i=1}^{k} (U_i + 1) \leq 2.$$

Direct Implications

From the schedulability condition $\left(\frac{C_k}{t_k}+\frac{\alpha}{\beta}\right)\prod_{j=1}^{k-1}(\beta U_j+1)\leq\frac{\alpha}{\beta}+1$

Lemma

(Lemma 2) Task τ_k is schedulable by the scheduling algorithm if

$$\frac{C_k}{t_k}+\sum_{i=1}^{k-1}U_i\leq\begin{cases}1, & (\alpha+\beta)^{\frac{1}{k}}<1\\ (k-1)\frac{\left(\left(1+\frac{\beta}{\alpha}\right)^{\frac{1}{k-1}}-1\right)}{\beta}, & (\alpha+\beta)^{\frac{1}{k}}<\alpha\\ \frac{(k-1)\left((\alpha+\beta)^{\frac{1}{k}}-1\right)+\left((\alpha+\beta)^{\frac{1}{k}}-\alpha\right)}{\beta} & \textit{otherwise.}\end{cases}$$

Lemma

(Lemma 3) Task τ_k is schedulable by the scheduling algorithm if

$$\beta\sum_{i=1}^{k-1}U_i\leq\ln\left(\frac{\frac{\alpha}{\beta}+1}{\frac{C_k}{t_k}+\frac{\alpha}{\beta}}\right).$$

k-Point Effective Schedulability Test: k^2Q

Definition

[Last Release Time Ordering] Let π be the last release time ordering assignment as a bijective function $\pi : hp(\tau_k) \rightarrow \{1, 2, \ldots, k-1\}$ to define the last release time ordering of task $\tau_j \in hp(\tau_k)$ in the window of interest.

Definition

A k-last-release effective schedulability test under a given ordering π of the $k-1$ higher priority tasks is a sufficient schedulability test of a fixed-priority scheduling policy by verifying the existence of $t_1 \leq t_2 \leq \cdots \leq t_{k-1} \leq t_k$ such that

$$C_k + \sum_{i=1}^{k-1} \alpha_i t_i U_i + \sum_{i=1}^{j-1} \beta_i C_i \leq t_j, \tag{1}$$

where $C_k > 0$, for $i = 1, 2, \ldots, k-1$, $\alpha_i > 0$, $U_i > 0$, $C_i \geq 0$, and $\beta_i > 0$ are dependent upon the setting of the task models and task τ_i.

Key Lemma

Lemma

[Lemma 5] For a given k-point last-release schedulability test of a scheduling algorithm, in which $0 < \alpha_i$, and $0 < \beta_i$ for any $i = 1, 2, \ldots, k-1$, $0 < t_k$, $\sum_{i=1}^{k-1} \alpha_i U_i \leq 1$, and $\sum_{i=1}^{k-1} \beta_i C_i \leq t_k$, task τ_k is schedulable by the fixed-priority scheduling algorithm if the following condition holds

$$\frac{C_k}{t_k} \leq 1 - \sum_{i=1}^{k-1} \alpha_i U_i - \frac{\sum_{i=1}^{k-1}(\beta_i C_i - \alpha_i U_i(\sum_{\ell=i}^{k-1} \beta_\ell C_\ell))}{t_k}. \qquad (2)$$

Key Lemma

Lemma

[Lemma 5] For a given k-point last-release schedulability test of a scheduling algorithm, in which $0 < \alpha_i$, and $0 < \beta_i$ for any $i = 1, 2, \ldots, k-1$, $0 < t_k$, $\sum_{i=1}^{k-1} \alpha_i U_i \leq 1$, and $\sum_{i=1}^{k-1} \beta_i C_i \leq t_k$, task τ_k is schedulable by the fixed-priority scheduling algorithm if the following condition holds

$$\frac{C_k}{t_k} \leq 1 - \sum_{i=1}^{k-1} \alpha_i U_i - \frac{\sum_{i=1}^{k-1}(\beta_i C_i - \alpha_i U_i(\sum_{\ell=i}^{k-1} \beta_\ell C_\ell))}{t_k}. \qquad (2)$$

The worst-case ordering π of the $k-1$ higher-priority tasks is to order the tasks in a non-increasing order of $\frac{\beta_i C_i}{\alpha_i U_i}$.

A Sketched Proof

The unschedulability implies that $C_k > C_k^*$, where C_k^* is defined in the following optimization problem:

$$
\begin{aligned}
\text{infimum} \quad & C_k^* \\
\text{such that} \quad & C_k^* + \sum_{i=1}^{k-1} \alpha_i t_i^* U_i + \sum_{i=1}^{j-1} \beta_i C_i > t_j^*, && \forall j = 1, 2, \ldots, k \\
& t_j^* \geq 0, && \forall j = 1, 2, \ldots, k,
\end{aligned}
$$

where $t_1^*, t_2^*, \ldots, t_{k-1}^*$ and C_k^* are variables, α_i, β_i are constants, and t_k^* is defined as t_k.

A Sketched Proof

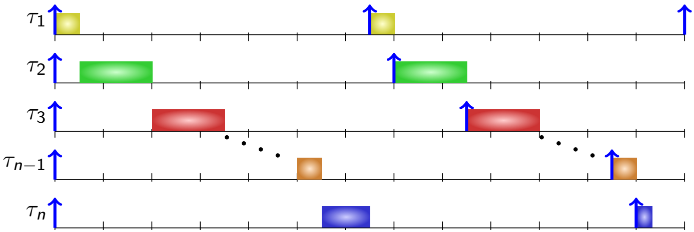
The unschedulability implies that $C_k > C_k^*$, where C_k^* is defined in the following optimization problem:

$$
\begin{aligned}
\text{infimum} \quad & C_k^* \\
\text{such that} \quad & C_k^* + \sum_{i=1}^{k-1} \alpha_i t_i^* U_i + \sum_{i=1}^{j-1} \beta_i C_i > t_j^*, && \forall j = 1, 2, \ldots, k \\
& t_j^* \geq 0, && \forall j = 1, 2, \ldots, k,
\end{aligned}
$$

where $t_1^*, t_2^*, \ldots, t_{k-1}^*$ and C_k^* are variables, α_i, β_i are constants, and t_k^* is defined as t_k.

The above linear programming gives the minimum C_k^* to be unschedulable. Therefore, if $C_k \leq C_k^*$, task τ_k is guaranteed to be schedulable.

Quadratic Bound for Sporadic Task Systems



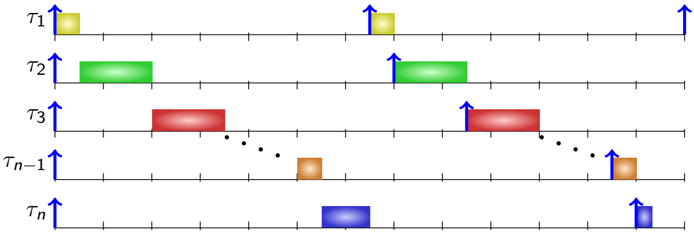
Let t_i be $\left\lfloor \frac{T_k}{T_i} \right\rfloor T_i$. Therefore, $\alpha_i = 1$ and $\beta_i \leq 1$.

Theorem

Task τ_k is schedulable by RM on a uniprocessor system if

$$0 \leq 1 - U_k - 2\sum_{i=1}^{k-1} U_i + 0.5\left((\sum_{i=1}^{k-1} U_i)^2 + (\sum_{i=1}^{k-1} U_i^2) \right).$$

Quadratic Bound for Sporadic Task Systems



Let t_i be $\left\lfloor \frac{T_k}{T_i} \right\rfloor T_i$. Therefore, $\alpha_i = 1$ and $\beta_i \leq 1$.

Theorem

Task τ_k is schedulable by RM on a uniprocessor system if

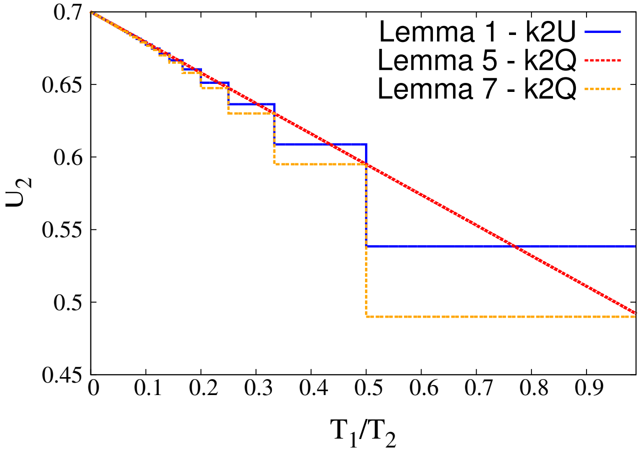
$$0 \leq 1 - U_k - 2\sum_{i=1}^{k-1} U_i + 0.5\left((\sum_{i=1}^{k-1} U_i)^2 + (\sum_{i=1}^{k-1} U_i^2)\right).$$

How to Use the Framework

- Parameters α_i and β_i affect the quality of the schedulability bounds
- Deriving the *good* settings of α_i and β_i is not part of this framework.
  - The framework simply derives the bounds/tests according to α_i and β_i
  - The correctness of α_i and β_i is not verified by the framework.
- The hyperbolic/quadratic bounds or utilization bounds can be automatically derived
  - The other approaches seek for the total utilization bounds
  - They have limited applications and are less flexible.
- After α_i and β_i or their safe upper bounds α and β are derived, the task model is not further referred.

Comparisons

Adopting different tests from k^2U and k^2Q for RM uniprocessor scheduling with $k = 2$ and $U_1 = 0.3$.



When to Use Which?

- k^2U (Version 1 above)
  - define any valid k points to obtain the corresponding α_i and β_i
  - more precise if the corresponding exponential-time (pseudo-polynomial-time) test is an exact test
  - may be less precise if the corresponding test requires some pessimism to be constructed, to be shown later
- k^2Q (Version 2 above)
  - define k last release points to obtain the corresponding α_i and β_i
  - has to typically consider the last release ordering
  - less precise if the corresponding exponential-time (pseudo-polynomial-time) test is an exact test
  - may be more precise by starting from the exponential-time test, to be shown later
  - can be generalized for response-time analysis

Outline

Introduction

Utilization-Based Analytical Framework

Selected Applications

Conclusions

Deadline-Monotonic Scheduling

- let $hp(\tau_k)$ be the set of tasks with higher priority than τ_k.
  - $hp_1(\tau_k)$ consists of the higher-priority tasks τ_i with $T_i < D_k$.
  - $hp_2(\tau_k)$ consists of the higher-priority tasks τ_i with $T_i \geq D_k$.
- The schedulability test is equivalent to the verification of

$$\exists 0 < t \leq D_k \qquad C_k + \sum_{\tau_i \in hp_2(\tau_k)} C_i + \sum_{\tau_i \in hp_1(\tau_k)} \left\lceil \frac{t}{T_i} \right\rceil C_i \leq t.$$

$$\Rightarrow \exists 0 < t \leq D_k \qquad C'_k + \sum_{\tau_i \in hp_1(\tau_k)} \left\lceil \frac{t}{T_i} \right\rceil C_i \leq t.$$

- Apply k^2U or k^2Q to get the utilization-based schedulability tests, by setting $\alpha_i = 1$ and $0 < \beta_i \leq 1$.

Deadline-Monotonic Scheduling (cont.)

The schedulability condition of task τ_k by using k^2U is

$$\left(\frac{C'_k}{D_k} + 1\right) \prod_{\tau_j \in hp_1(\tau_k)} (U_j + 1) \leq 2.$$

The schedulability condition of task τ_k by using k^2Q is

$$\frac{C'_k}{D_k} \leq 1 - 2\sum_{i=1}^{k-1} U_i + 0.5\left((\sum_{i=1}^{k-1} U_i)^2 + (\sum_{i=1}^{k-1} U_i^2)\right).$$

- It can be proved that the speed-up factor of DM is 1.76322, compared to EDF.

Uniprocessor Self-Suspending Task Systems

For all $0 < t \leq T_k$

$$W_k(t) = \sum_{i=1}^{k-1} \left(\left\lceil \frac{t}{T_i} \right\rceil - 1 \right) C_i + 2C_i.$$

Schedulability test for task τ_k:

$$\exists t \text{ with } 0 < t \leq T_k \;\; \textit{and} \;\; C_k + S_k + W_k(t) \leq t.$$

Uniprocessor Self-Suspending Task Systems

For all $0 < t \leq T_k$

$$W_k(t) = \sum_{i=1}^{k-1} \left(\left\lceil \frac{t}{T_i} \right\rceil - 1 \right) C_i + 2C_i.$$

Schedulability test for task τ_k:

$$\exists t \text{ with } 0 < t \leq T_k \;\; and \;\; C_k + S_k + W_k(t) \leq t.$$

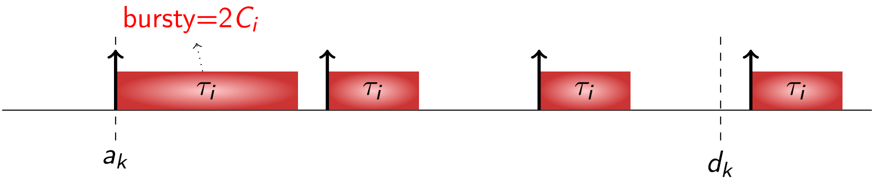
Uniprocessor Self-Suspending Task Systems

For all $0 < t \leq T_k$

$$W_k(t) = \sum_{i=1}^{k-1} \left(\left\lceil \frac{t}{T_i} \right\rceil - 1 \right) C_i + 2C_i.$$

Schedulability test for task τ_k:

$$\exists t \text{ with } 0 < t \leq T_k \;\; and \;\; C_k + S_k + W_k(t) \leq t.$$



Uniprocessor Self-Suspending: k2U Framework

- Let t_i be $\left\lfloor \frac{T_k}{T_i} \right\rfloor T_i$
- When testing task τ_k,
  - $\alpha_i \leq \alpha = 2$ and $\beta_i \leq \beta = 1$ for i=1,2,. . . ,k-1
- By using k^2U framework, τ_k is schedulable by RM scheduling if

$$\left(\frac{C_k + S_k}{T_k} + 2\right) \prod_{j=1}^{k-1} (U_j + 1) \leq 3.$$

In the key lemma:

$$\left(\frac{C_k}{t_k} + \frac{\alpha}{\beta}\right) \prod_{j=1}^{k-1} (\beta U_j + 1) \leq \frac{\alpha}{\beta} + 1$$

Utilization Bounds

Let $t_i = \left\lfloor \frac{T_i}{T_k} \right\rfloor T_i$. Therefore, we have $\alpha_i \leq 2$ and $\beta_i \leq 1$.

Theorem (Liu and Chen in RTSS 2014)

Any implicit-deadline sporadic self-suspending task set is schedulable under RM if the following conditions hold:

$$\forall 1 \leq k \leq n, \boxed{U_k + \frac{S_k}{T_k}} \leq 1 - (\boxed{2} + 1) \cdot \left(1 - \frac{1}{\prod_{i=1}^{k-1}(U_i + 1)}\right). \quad (3)$$

Theorem (Liu and Chen in RTSS 2014)

Any implicit-deadline sporadic self-suspending task set is schedulable under RM if the following conditions hold:

$$\forall 1 \leq k \leq n, U_k + \boxed{\frac{S_k}{T_k}} + \sum_{i=1}^{k-1} U_i \leq k \left(\left(\frac{\boxed{2} + 1}{\boxed{2}}\right)^{\frac{1}{k}} - 1 \right) \quad (4)$$

Uniprocessor Non-Preemptive (NP) Scheduling

Let $\widehat{C_k} = C_k + B_k + \sum_{\tau_i \in hp_2(\tau_k)} C_i$, where B_k is $\left\{\max_{\tau_i \in lp(\tau_k)} C_i\right\}$. The schedulability condition of task τ_k by using k^2U is

$$\left(\frac{\widehat{C'_k}}{D_k} + 1\right) \prod_{\tau_j \in hp_1(\tau_k)} (U_j + 1) \leq 2 \tag{5}$$

Theorem

[Theorem 4 in von der Brüeggen, Chen, and Huang, 2015]
Suppose that $\gamma = \max_{\tau_k} \left\{\max_{\tau_i \in lp(\tau_k)} \left\{\frac{C_i}{C_k}\right\}\right\}$. A task set can be feasibly scheduled by RM-NP if

$$U_{sum} \leq \begin{cases} \frac{\gamma}{1+\gamma} + ln\left(\frac{2}{1+\gamma}\right) & \text{if } \gamma \leq 1 \\ \frac{1}{1+\gamma} & \text{if } \gamma > 1 \end{cases}$$

(left as an exercise)

Real-Time Systems with Mode Changes

- Real-time tasks run in different modes over time to react to the change of physical environments
  - Avionic systems
  - Automotive systems

rotation (rpm)	functions to be executed
$[0, 2000]$	$f1(); f2(); f3(); f4();$
$(2000, 4000]$	$f1(); f2(); f3();$
$(4000, 6000]$	$f1(); f2();$
$(6000, 8000]$	$f1();$

Multi-Mode Task Model

- A multi-mode task τ_i is denoted by a set of triplets:

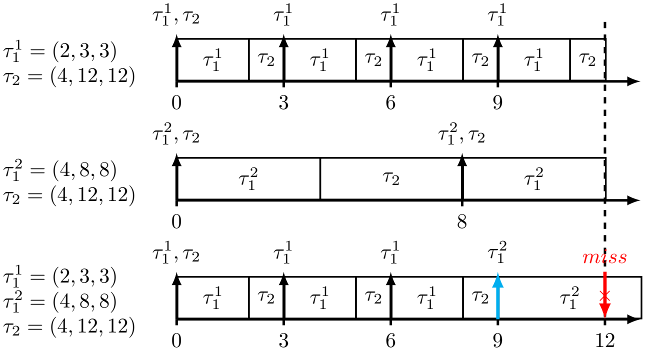
$$\begin{aligned}\tau_i = \{\tau_i^1 &= (C_i^1, T_i^1, D_i^1),\\ \tau_i^2 &= (C_i^2, T_i^2, D_i^2), \ldots,\\ \tau_i^{M_i} &= (C_i^{M_i}, T_i^{M_i}, D_i^{M_i})\}\end{aligned}$$

  - C_i^m denotes the *worst-case execution time* (WCET) of task τ_i under mode m
  - T_i^m denotes the *minimum inter-arrival time* of task τ_i under mode m
  - D_i^m denotes relative deadline (Constrained-deadline system ($D_i^m \leq T_i^m$))
- Fixed-priority scheduling
  - Fixed-priority task-level (FPT)
  - Fixed-priority mode-level (FPM)

Problems with a Naive Analysis

Deadline miss during mode transition under fixed-priority scheduling

- $\tau_1 = \{\tau_1^1 = (2, 3, 3), \tau_1^2 = (4, 8, 8)\}$
- $\tau_2 = (4, 12, 12)$



A Safe Exponential-Time Test

Theorem

Task mode τ_k^h is schedulable under an FPM scheduling if

$$\forall y \geq 0, \ \ \forall \textit{combinations of } t_i^* \textit{ with } 0 \leq t_i^* \leq t_{i+1}^*, \forall i = 1, 2, \ldots, k-1$$

$$\exists j = 1, 2, \ldots, k, s.t. \ \ C_k^h + y \cdot U_k^{max} + \sum_{i=1}^{k-1} (U_i^{max} \cdot t_i^*) + \sum_{i=1}^{j-1} C_i^{max} \leq t_j^*.$$

C_i^{max} is the maximum execution time among the modes of task τ_i with priority higher than or equals to task mode τ_k^h. U_i^{max} is the maximum utilization among the modes of task τ_i with priority higher than or equals to task mode τ_k^h. The constant t_k^* is defined as $T_k^h + y$.

A Safe Exponential-Time Test

Theorem

Task mode τ_k^h is schedulable under an FPM scheduling if

$$\forall y \geq 0,\ \ \forall combinations\ of\ t_i^*\ with\ 0 \leq t_i^* \leq t_{i+1}^*, \forall i = 1, 2, \ldots, k-1$$

$$\exists j = 1, 2, \ldots, k, s.t.\ \ C_k^h + y \cdot U_k^{max} + \sum_{i=1}^{k-1} (U_i^{max} \cdot t_i^*) + \sum_{i=1}^{j-1} C_i^{max} \leq t_j^*.$$

C_i^{max} is the maximum execution time among the modes of task τ_i with priority higher than or equals to task mode τ_k^h. U_i^{max} is the maximum utilization among the modes of task τ_i with priority higher than or equals to task mode τ_k^h. The constant t_k^* is defined as $T_k^h + y$.

Hint: the above test also requires to enumerate all possible orderings. Under k^2Q, it is possible to safely only test one specific ordering.

Utilization Test under FPM-RM

For a given y, we have $C_k^h + y \cdot U_k^{max} \leq (T_k + y) \cdot U_k^{max}$. So, the remaining is a case with $\alpha_1 = 1$ and $\beta_i = 1$ in the k^2Q framework.

Theorem

Task τ_k^h in a multi-mode task system with implicit deadlines is schedulable by the mode-level RM scheduling algorithm on a uniprocessor system if the following condition holds

$$U_k^{\max} \leq 1 - 2\sum_{i=1}^{k-1} U_i^{\max} + 0.5\left((\sum_{i=1}^{k-1} U_i^{\max})^2 + (\sum_{i=1}^{k-1} (U_i^{\max})^2) \right), \tag{6}$$

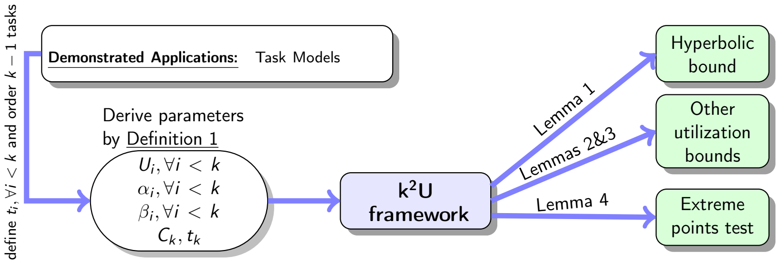
or

$$\sum_{i=1}^{k-1} U_i^{\max} \leq \left(\frac{k-1}{k}\right)\left(2 - \sqrt{2 + 2U_k^{\max}\frac{k}{k-1}}\right), \tag{7}$$

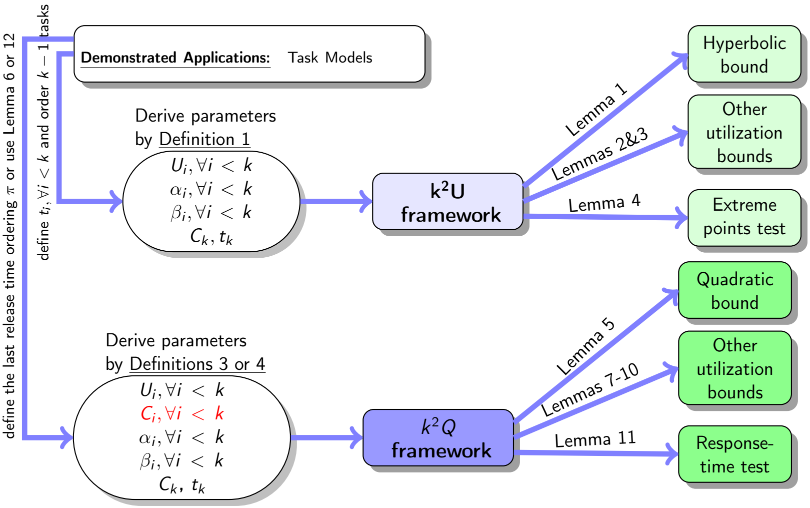
or

$$U_k^{\max} + \sum_{i=1}^{k-1} U_i^{\max} \leq \begin{cases} \left(\frac{k-1}{k}\right)\left(2 - \sqrt{4 - \frac{2k}{k-1}}\right), & \text{if } k > 3 \\ 1 - \frac{(k-1)}{2k} & \text{otherwise.} \end{cases} \tag{8}$$

Framework


define $t_i, \forall i < k$ and order $k-1$ tasks
Demonstrated Applications: Task Models
Derive parameters by Definition 1
$U_i, \forall i < k$
$\alpha_i, \forall i < k$
$\beta_i, \forall i < k$
C_k, t_k
k^{2}U framework
Lemma 1
Lemmas 2&3
Lemma 4
Hyperbolic bound
Other utilization bounds
Extreme points test

Framework



define the last release time ordering π or use Lemma 6 or 12
define $t_i, \forall i < k$ and order $k-1$ tasks
Demonstrated Applications: Task Models
Derive parameters by Definition 1
$U_i, \forall i < k$
$\alpha_i, \forall i < k$
$\beta_i, \forall i < k$
C_k, t_k
k²U framework
Lemma 1
Lemmas 2&3
Lemma 4
Hyperbolic bound
Other utilization bounds
Extreme points test
Derive parameters by Definitions 3 or 4
$U_i, \forall i < k$
$C_i, \forall i < k$
$\alpha_i, \forall i < k$
$\beta_i, \forall i < k$
C_k, t_k
k^2Q framework
Lemma 5
Lemmas 7-10
Lemma 11
Quadratic bound
Other utilization bounds
Response-time test

Outline

Introduction

Utilization-Based Analytical Framework

Selected Applications

Conclusions

Polynomial-time schedulability test framework

- Basically handles many problems very well with low time complexity
- The first evidence to translate *exponential-time schedulability tests* to (potentially) *linear-time tests* with test quality guarantees

Polynomial-time schedulability test framework

- Basically handles many problems very well with low time complexity
- The first evidence to translate *exponential-time schedulability tests* to (potentially) *linear-time tests* with test quality guarantees

- Can we derive the coefficients α_i and β_i *automatically?*
  - Yes, for some well-studied forms. See our recent technical report. One commonly used class as an example:

$$\exists 0 < t \leq D_k \text{ s.t. } C_k + \sum_{i=1}^{k-1} \sigma \left(\left\lceil \frac{t}{T_i} \right\rceil C_i + bC_i \right) \leq t.$$

  - No idea yet for arbitrary forms

Other Applications

- Multi-frame task model
- Digraph task models
- Uniprocessor/Multiprocessor scheduling with self-suspensions
- Multiprocessor global DM scheduling
- Multiprocessor partitioned RM/DM scheduling
- Multiprocessor scheduling with DAG structures
- etc.