Stability Conditions of On-line Resource Managers for Systems with Execution Times Variations

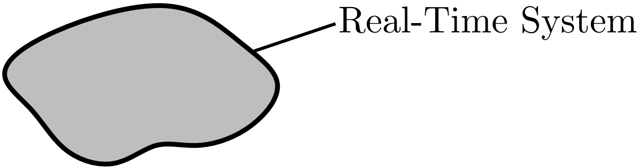
Sergiu Rafiliu, Petru Eles, Zebo Peng

Department of Computer and Information Science,
Linköping University, Sweden

July 29, 2011

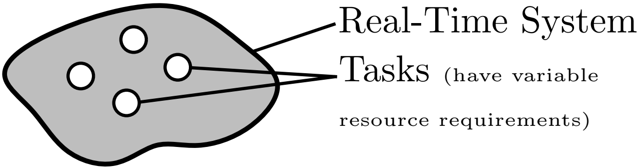
Preliminaries

Stability Conditions of On-line Resource Managers for Systems with Execution Times Variations



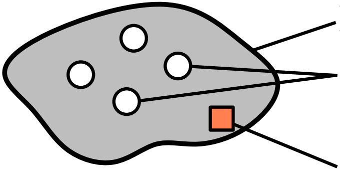
Preliminaries

Stability Conditions of On-line Resource Managers for Systems with Execution Times Variations



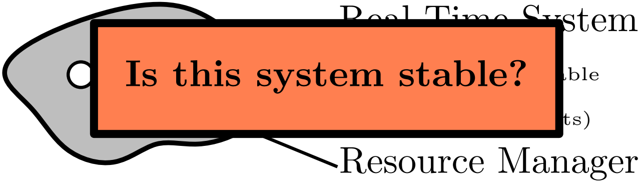
Preliminaries

Stability Conditions of On-line Resource Managers for Systems with Execution Times Variations

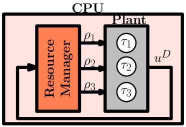


Preliminaries

Stability Conditions of On-line Resource Managers for Systems with Execution Times Variations



Motivation

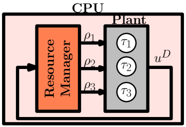


On-line Resource Managers

- React to resource demand (variable execution times)
- Control the state of the system (response times, latency, jitter, throughput, load, . . .).
- Other goals (e.g. Optimize some performance QoS metric)

Stability means that the Resource Manager controls the system such that the resource demand (state) is bounded.

Motivation



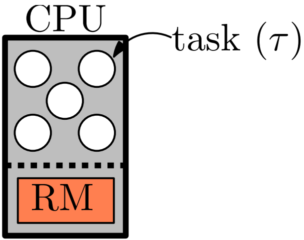
On-line Resource Managers

- React to resource demand (variable execution times)
- Control the state of the system (response times, latency, jitter, throughput, load, . . .).
- Other goals (e.g. Optimize some performance QoS metric)

Stability means that the Resource Manager controls the system such that the resource demand (state) is bounded.

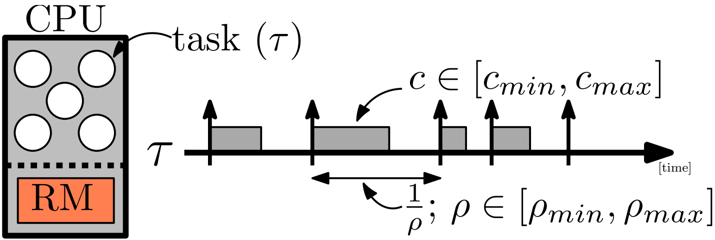
Outline

System Description



- Execution times (c) vary in unknown ways
- Task rates (ρ) are decided by the **Resource Manager** (RM)
- Jobs are scheduled through any scheduling policy that:
  - executes the jobs of each task in the order of their release
  - is non-idling.

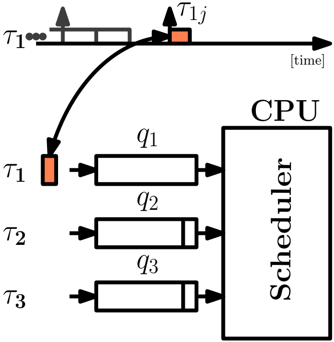
System Description



- Execution times (c) vary in unknown ways
- Task rates (ρ) are decided by the **Resource Manager** (RM)
- Jobs are scheduled through any scheduling policy that:
  - executes the jobs of each task in the order of their release
  - is non-idling.

System Description

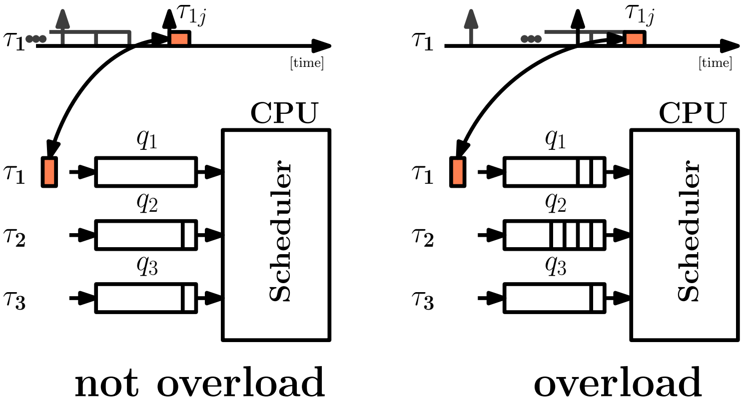
Tasks are modeled as queues of jobs, that wait to be executed by the scheduler.



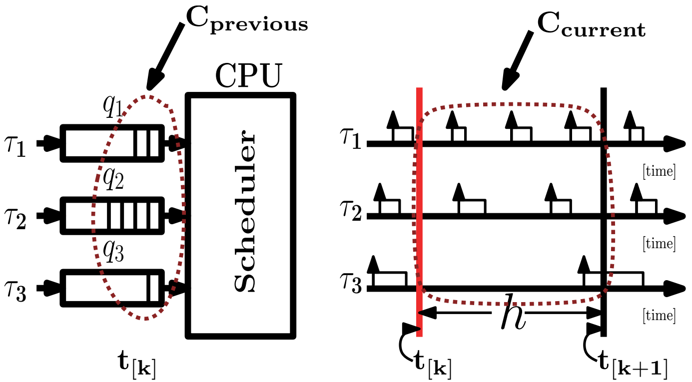
not overload

System Description

Tasks are modeled as queues of jobs, that wait to be executed by the scheduler.



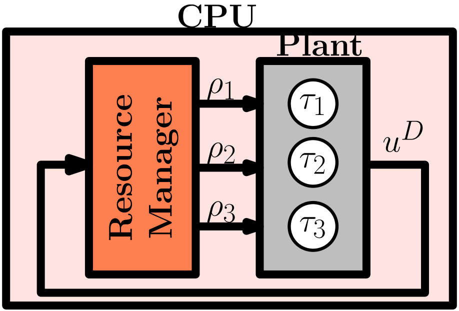
Resource Demand



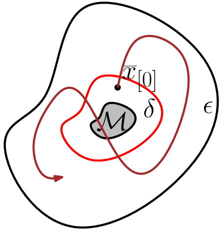
$$\text{(predicted)}\mathbf{u}^{D}_{[k]} = \frac{C_{previous} + C_{current}}{t_{[k+1]} - t_{[k]}}$$

Resource Manager

- Runs at key moments in time: $t_{[k]}, t_{[k+1]}, t_{[k+2]}, \cdots$
- Reacts to variations in resource demand (u^D),
- Adjusts task rates (ρ_i) to new values.



Stability - Definition (from Control Theory)



Definition (Stability in the sense of Lagrange)

A system is stable, if at any moment in time, the system's state is within a bounded distance from the set of stationary points.

Stationary Points are states where we wish our system to spend most of its time. In our case, any state where the ***resource demand (u^D) is*** **1** is a stationary point.

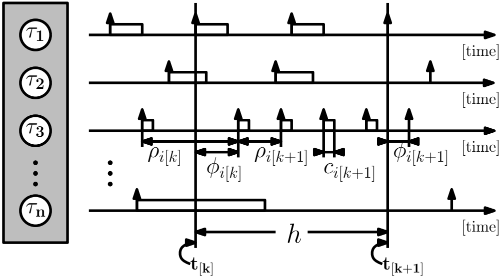
Problem Formulation

Determine a criterion that the Resource Manager must satisfy, in order for the system to be stable.

System Model

System of difference equations

$$F(\overline{x}_{[k+1]}, \overline{x}_{[k]}) = 0$$



$$\overline{x}_{[k]} = \begin{pmatrix} \overline{\Phi_{[k]}} \\ \overline{\rho_{[k]}} \\ \overline{c_{[k]}} \\ \overline{q_{[k]}} \\ \vdots \end{pmatrix} \begin{matrix} \leftarrow \text{offsets} \\ \leftarrow \text{rates} \\ \leftarrow \text{predicted execution times} \\ \leftarrow \text{queue sizes} \\ \vdots \end{matrix}$$

System Model

$$\sum_{i \in I} c_{i[k+1]} \cdot q_{i[k+1]} = h \cdot \max\{0, u^R_{[k+1]} - 1\}$$

$$\phi_{i[k+1]} = \phi_{i[k]} + \frac{1}{\rho_{i[k+1]}} \left\lceil \rho_{i[k+1]} \cdot \max\{0, h - \phi_{i[k]}\} \right\rceil - h$$

$$\overline{c}_{[k+1]} = f_p(c_{i[k]}) + \nu_{[k]}$$

$$\overline{q^{\mathrm{p}}}_{[k+1]} = \overline{q}_{[k]}$$

$$\overline{\phi^{\mathrm{p}}}_{[k+1]} = \overline{\phi}_{[k]}$$

$$\overline{\rho}_{[k+1]} = f_c(\overline{x}_{[k]})$$

System Model

- resource manager

$$\overline{\rho}_{[k+1]} = f_c(\overline{x}_{[k]})$$

- execution time prediction

$$\overline{c}_{[k+1]} = f_p(\overline{x}_{[k]}) + \overline{\nu}_{[k]}$$

- model of the scheduling policy

$$\sum_{i \in I} c_{i[k+1]} \cdot q_{i[k+1]} = h \cdot \max\{0, u^R_{[k+1]} - 1\}$$

Stability - Intuition

$$\overline{x}_{[k]} = \begin{pmatrix} \overline{\Phi_{[k]}} \\ \overline{\rho_{[k]}} \\ \overline{c_{[k]}} \\ \overline{q_{[k]}} \\ \vdots \end{pmatrix} \begin{array}{l} \left.\begin{array}{l} \leftarrow \text{offsets} \\ \leftarrow \text{rates} \\ \leftarrow \text{predicted execution times} \end{array}\right\} \text{bounded above and below} \\ \leftarrow \text{queue sizes} \;\} \textbf{unbounded above} \\ \vdots \end{array}$$

Stability - Criterion

Theorem 2 (Necessary condition)

A necessary condition for a system to be stable is:

$$\sum_{i \in I} c_i^{\max} \cdot \rho_i^{\min} \leq 1$$

- All jobs of all tasks could have their worst-case execution times
- There must exist a setting for task rates, such that, in this conditions, the load does not exceed 1

Stability - Criterion

Theorem 2 (Necessary condition)

A necessary condition for a system to be stable is:

$$\sum_{i \in I} c_i^{\max} \cdot \rho_i^{\min} \leq 1$$

The set of all rate vectors that guarantee that the resource demand drops is: $\Gamma_\star = \left\{ \overline{\rho} \in \mathbf{P} \middle| \sum_{i \in I} c_i^{\max} \cdot \rho_i \leq 1 \right\}$

Stability - Criterion

Theorem 3

For any general system that satisfies Theorem 2 a sufficient stability condition is that the Resource Manager satisfies:

$$\overline{\rho}_{[k+1]} \in \begin{cases} \Gamma_\star, & \text{if } u^D_{[k]} \geq u^D_\Omega \\ \mathbf{P}, & \text{otherwise} \end{cases} \tag{1}$$

$$\rho_{i[k+1]} \leq \rho_{i[k]} \quad \text{if } u^D_{[k]} \geq u^D_\Omega, \quad \forall i \in I \tag{2}$$

Stability Criterion – Proof

Theorem 1 (Uniform boundedness)

A system is uniformly bounded if:

- $d(\overline{x}_{[k]}, \mathcal{M}) \geq \Omega \Rightarrow \exists V : \mathcal{X} \rightarrow \mathbb{R}_+$ and $\varphi_1, \varphi_2 : \mathbb{R}_+ \rightarrow \mathbb{R}_+; \varphi_1, \varphi_2 \uparrow$, $\lim_{r\rightarrow\infty} \varphi_i(r) = \infty, i = \overline{1,2}$

$$\varphi_1(d(\overline{x}, \mathcal{M})) \leq V(\overline{x}) \leq \varphi_2(d(\overline{x}, \mathcal{M})) \text{ and}$$
$$V(\overline{x}_{[k+1]}) \leq V(\overline{x}_{[k]})$$

- $d(\overline{x}_{[k]}, \mathcal{M}) < \Omega \Rightarrow$

$$\exists \Psi > 0 \text{ s.t } d(\overline{x}_{[k+1]}, \mathcal{M}) < \Psi$$

$u^D_\Omega \Rightarrow \Omega, \Psi$
$u^D(\overline{x}) \Rightarrow V(\overline{x})$ (upper bound)

Ω – ultimate bound
Ψ – bound on the real-time performance of the system

Stability Criterion – Proof

Theorem 1 (Uniform boundedness)

A system is uniformly bounded if:

- $d(\overline{x}_{[k]}, \mathcal{M}) \geq \Omega \Rightarrow \exists V : \mathcal{X} \to \mathbb{R}_+$ and $\varphi_1, \varphi_2 : \mathbb{R}_+ \to \mathbb{R}_+; \varphi_1, \varphi_2 \uparrow$, $\lim_{r\to\infty} \varphi_i(r) = \infty, i = \overline{1,2}$

$$\boxed{\varphi_1(d(\overline{x}, \mathcal{M}))} \leq \boxed{V(\overline{x})} \leq \boxed{\varphi_2(d(\overline{x}, \mathcal{M}))} \text{ and}$$

$$V(\overline{x}_{[k+1]}) \leq V(\overline{x}_{[k]})$$

- $d(\overline{x}_{[k]}, \mathcal{M}) < \Omega \Rightarrow$

$$\exists \Psi > 0 \text{ s.t } d(\overline{x}_{[k+1]}, \mathcal{M}) < \Psi$$

$u^D_\Omega \Rightarrow \Omega, \Psi$

$u^D(\overline{x}) \Rightarrow V(\overline{x})$ (upper bound)

Ω – ultimate bound

Ψ – bound on the real-time performance of the system

Usage

Three ways:

- Select u_{Ω}^{D} and construct a resource manager,
- Take an existing resource manager and show that there exists a u_{Ω}^{D}, or
- Modify an existing resource manager by considering a u_{Ω}^{D}

$u_{\Omega}^{D} \Rightarrow \Psi$ – bound on the real-time performance of the system.

Stability of Existing Resource Manager

- QRAM[1]
- Corner-Case[2]
- QoS Derivative[2]

[1] C. Lee, J. Lehoczky, R. Rajkumar, D. Siewiorek. *"On Quality of Service Optimization with Discrete QoS Options."* In proceedings of Real-Time Technology and Applications Symposium, pp.276, 1999.

[2] S. Rafiliu, P. Eles, and Z. Peng. *"Low Overhead Dynamic QoS Optimization Under Variable Execution Times."* Proceedings of 16th IEEE Embedded and Real-Time computing Systems and Applications, pp. 293-303, 2010.

Stability of QRAM

Each time QRAM runs, it takes the following steps:

- set all rates to $\rho_i^{\min}$ and compute u^D with the assumed rates.
- if $u^D < u_\Omega^D$ select a tasks τ_i according with some given QoS curves
- increase τ_i's rate to the next defined rate point on the curve, or until $u^D = u_\Omega^D$
- repeat this process until $u^D = u_\Omega^D$ or the rates of all tasks are $\rho_i^{\max}$

Observation

$u^D > u_\Omega^D \quad \Leftrightarrow \quad \overline{\rho} = (\rho_1^{min} \rho_2^{min} \rho_3^{min} \cdots)^T \in \Gamma_\star \quad \Rightarrow$ the system is stable.

Stability of QRAM

Each time QRAM runs, it takes the following steps:

- set all rates to $\rho_i^{\min}$ and compute u^D with the assumed rates.
- if $u^D < u_\Omega^D$ select a tasks τ_i according with some given QoS curves
- increase τ_i's rate to the next defined rate point on the curve, or until $u^D = u_\Omega^D$
- repeat this process until $u^D = u_\Omega^D$ or the rates of all tasks are $\rho_i^{\max}$

Observation

$u^D > u_\Omega^D \quad \Leftrightarrow \quad \bar{\rho} = (\rho_1^{min} \rho_2^{min} \rho_3^{min} \cdots)^T \in \Gamma_\star \quad \Rightarrow$ the system is stable.

Bounds for Stable Systems

Assume

- EDF scheduler
- Stable system, with $u^D_{[k]} \leq u^D_\Omega$

Results

- Bound on queue sizes:

$$q_j^{\max} = \frac{h}{c_j^{\min}} \cdot u^R_{\max}(u^D_\Omega)$$

- Bound on response times:

$$r_j^{\max} = \frac{1}{\rho_j^{\min}} + h \cdot \left(u^R_{\max}(u^D_\Omega) - \frac{1}{h} \sum_{i \in I} c_i^{\max} \right) + \sum_{i \in I} c_i^{\max} \cdot \left\lfloor \frac{\rho_i^{\max}}{\rho_j^{\min}} \right\rfloor$$

Conclusions

- Proposed a stability criterion for a uniprocessor system with independent task set.
- Shown how to apply the criterion for a number of existing Resource Managers.
- Shown how the stability criterion can be linked with real-time properties (e.g. bound on execution times).

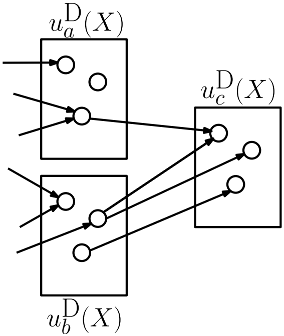
Current/Future Work

- more classes of schedulers
- tasks with different task modes
- allow job dropping
- more flexible release method
- different real-time goals

Current/Future Work

Distributed architectures composed of

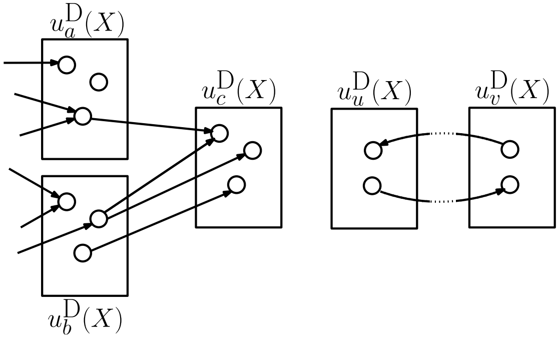
- several resources (CPUs, buses, . . .)
- task graphs distributed on the resources



Current/Future Work

Distributed architectures composed of

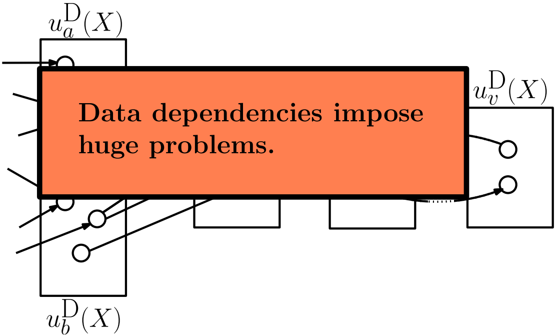
- several resources (CPUs, buses, . . .)
- task graphs distributed on the resources



Current/Future Work

Distributed architectures composed of

- several resources (CPUs, buses, . . .)
- task graphs distributed on the resources



Data dependencies impose huge problems.

Questions???

Stability - Theorem

Theorem 1 (Uniform boundedness)

A system is uniformly bounded if:

- $d(\overline{x}_{[k]}, \mathcal{M}) \geq \Omega \Rightarrow \exists V : \mathcal{X} \rightarrow \mathbb{R}_+$ and $\varphi_1, \varphi_2 : \mathbb{R}_+ \rightarrow \mathbb{R}_+; \varphi_1, \varphi_2 \uparrow$, $\lim_{r\rightarrow\infty} \varphi_i(r) = \infty, i = \overline{1,2}$

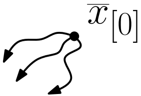
$$\begin{aligned} \varphi_1(d(\overline{x}, \mathcal{M})) \leq V(\overline{x}) \leq \varphi_2(d(\overline{x}, \mathcal{M})) \text{ and} \\ V(\overline{x}_{[k+1]}) \leq V(\overline{x}_{[k]}) \end{aligned}$$

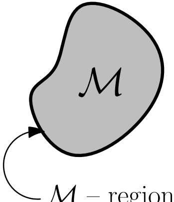
- $d(\overline{x}_{[k]}, \mathcal{M}) < \Omega \Rightarrow$

$$\exists \Psi > 0 \text{ s.t } d(\overline{x}_{[k+1]}, \mathcal{M}) < \Psi \tag{3}$$

Stability - Intuition

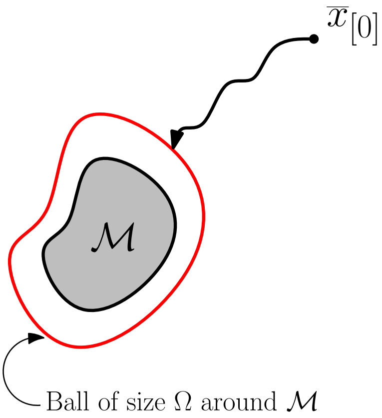
$\overline{x}_{[0]}$ – initial state



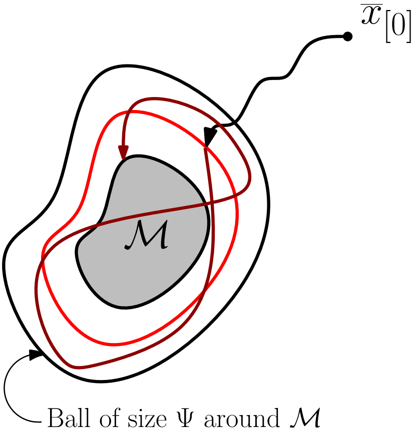


$\mathcal{M}$ – region of **stationary points**

Stability - Intuition



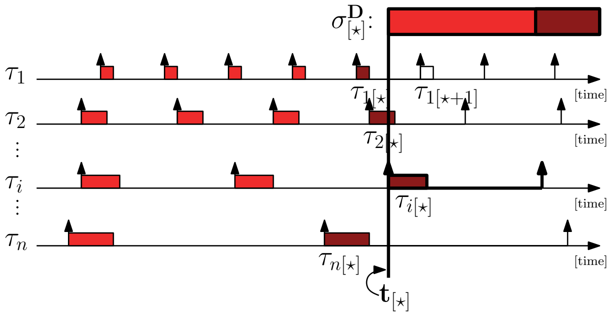
Stability - Intuition



Comparison of Models

Model	Controller	Example 1			Example 2		
		h	$r_1^{\max}$	$r_2^{\max}$	h	$r_1^{\max}$	$r_2^{\max}$
Constrained	**C1**	20	48	53	5000	91200	97050
	C2	20	28	33	5000	46200	52050
		h	$r_1^{\max}$	$r_2^{\max}$	h	$r_1^{\max}$	$r_2^{\max}$
General	**C1**	4	19	24	1000	19750	25600
	C2	4	12	17	1000	10200	16050
	C1	2	15	20	100	3550	9400
	C2	2	10	15	100	2100	7950

Response Time Result



$$r_j^{\max} = \frac{1}{\rho_j^{\min}} + h \cdot \left(u^R_{\max}(u^D_\Omega) - \frac{1}{h}\sum_{i \in I} c_i^{\max} \right) + \sum_{i \in I} c_i^{\max} \cdot \left\lfloor \frac{\rho_i^{\max}}{\rho_j^{\min}} \right\rfloor$$