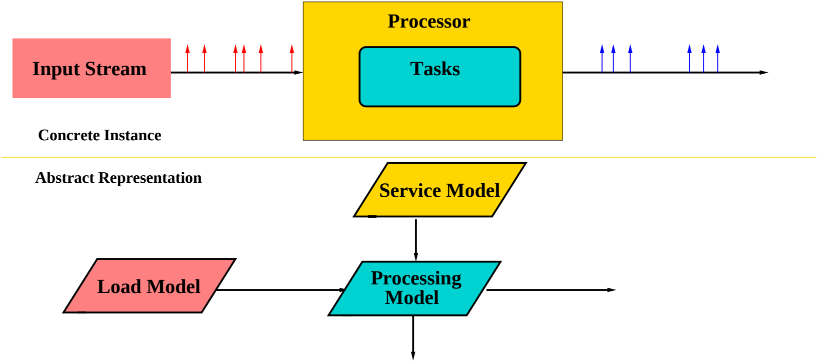
Real-Time Calculus and Module Performance Analysis

Prof. Dr. Jian-Jia Chen

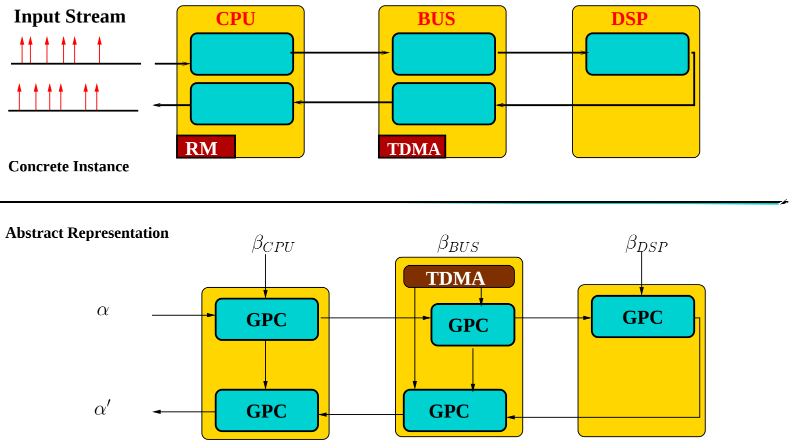
LS 12, TU Dortmund

14 June 2016

Abstract Models for Real-Time Calculus



Abstract Models for Module Performance Analysis



Overview

System View	Module Performance Analysis (MPA)
Math. View	Real-Time Calculus (RTC)
	Min-Plus Calculus, Max-Plus Calculus

Backgrounds

- Real-Time Calculus can be regarded as a worst-case/best-case variant of classical queuing theory. It is a formal method for the analysis of distributed real-time embedded systems.
- Related Work:
  - Min-Plus Algebra: F. Baccelli, G. Cohen, G. J. Olster, and J. P. Quadrat, Synchronization and Linearity - An Algebra for Discrete Event Systems, Wiley, New York, 1992.
  - Network Calculus: J.-Y. Le Boudec and P. Thiran, Network Calculus - A Theory of Deterministic Queuing Systems for the Internet, Lecture Notes in Computer Science, vol. 2050, Springer Verlag, 2001.

Plus-Times and Min-Plus Algebras

- Algebraic structure
  - a set of (finite or infinite) elements S
  - one or more operators defined on the elements of this set
- Plus-Times Algebra: Two operators $+$ and $\times$, denoted by $(S, +, \times)$
- Min-Plus Algebra
  - Two operators $\oplus$ (min) and $\otimes$ (plus), denoted by $(S \cup \{+\infty\}, \inf, +)$
  - Infimum:
    - The infimum of a subset of some set is the greatest element, not necessarily in the subset, that is less than or equal to all other elements of the subset.
    - For example, $\inf\{[a, b]\} = a$, $\inf\{(a, b]\} = a$, where $\min\{[a, b]\} = a$, $\min\{(a, b]\} =$ undefined.
  - Supremum:
    - The supremum of a subset of some set is the smallest element, not necessarily in the subset, that is more than or equal to all other elements of the subset.
    - For example, $\sup\{[a, b]\} = b$, $\sup\{[a, b)\} = b$, where $\max\{[a, b]\} = b$, $\max\{[a, b)\} =$ undefined.

Min-Plus Algebra: Properties for $\otimes$

Suppose that $a, b, c \in S$. We have

- Closure: $a \otimes b \in S$
- Associativity: $a \otimes (b \otimes c) = (a \otimes b) \otimes c$
- Commutativity: $a \otimes b = b \otimes a$
- Existence of identity element: $\exists \nu : a \otimes \nu = a$.
- Existence of negative element: $\exists a^{-1} : a \otimes a^{-1} = \nu$.
- Distributivity of $\otimes$ with respect to $\oplus$:
  $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$

Examples

- plus-times: $a \times (b + c) = a \times b + a \times c$
- min-plus: $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c) = \inf\{a + b, a + c\}$.

Min-Plus Algebra: Properties for $\oplus$

Suppose that $a, b, c \in S$. We have

- Closure: $a \oplus b \in S$
- Associativity: $a \oplus (b \oplus c) = (a \oplus b) \oplus c$
- Commutativity: $a \oplus b = b \oplus a$
- Existence of identity element: $\exists \epsilon : a \oplus \epsilon = a$.
- Property of ϵ regarding $\otimes$: $a \otimes \epsilon = \epsilon$.

Examples:

- plus-times: $\exists 0 : a + 0 = a$.
- min-plus: $a \oplus a = a$.

Definition of Arrival Curves and Service Curves

- For a specific trace:
  - Data streams: $R(t) =$ number of events in $[0, t)$
  - Resource stream: $C(t) =$ available resource in $[0, t)$
- For the worst cases and the best cases in any interval with length Δ:
  - Arrival Curve $[\alpha^l, \alpha^u]$:

$$\alpha^l(\Delta) = \inf_{\lambda \geq 0, \forall R} \{R(\Delta + \lambda) - R(\lambda)\}$$

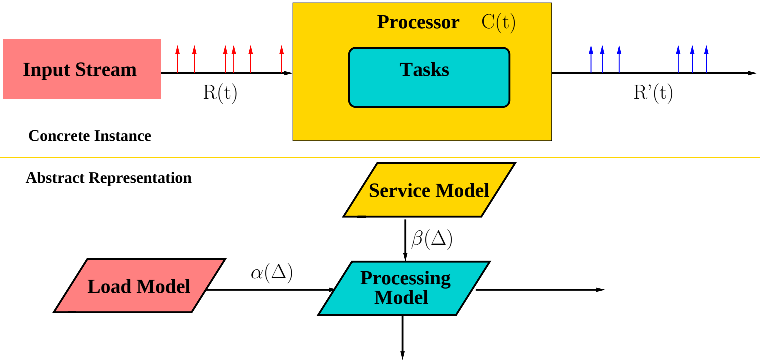
$$\alpha^u(\Delta) = \sup_{\lambda \geq 0, \forall R} \{R(\Delta + \lambda) - R(\lambda)\}$$

  - Service Curve $[\beta^l, \beta^u]$:

$$\beta^l(\Delta) = \inf_{\lambda \geq 0, \forall C} \{C(\Delta + \lambda) - C(\lambda)\}$$

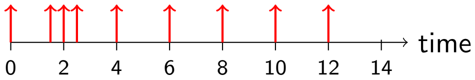
$$\beta^u(\Delta) = \sup_{\lambda \geq 0, \forall C} \{C(\Delta + \lambda) - C(\lambda)\}$$

Abstract Models for Real-Time Calculus



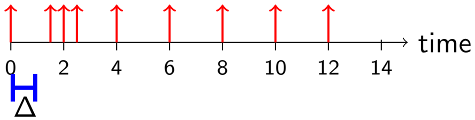
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



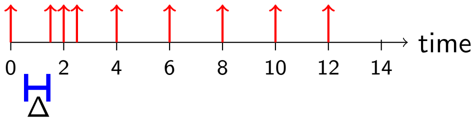
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



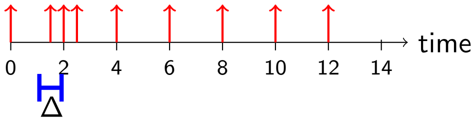
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



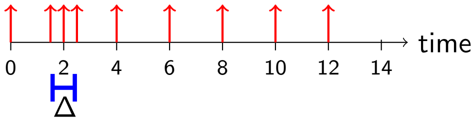
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



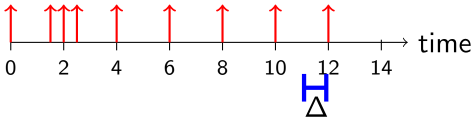
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



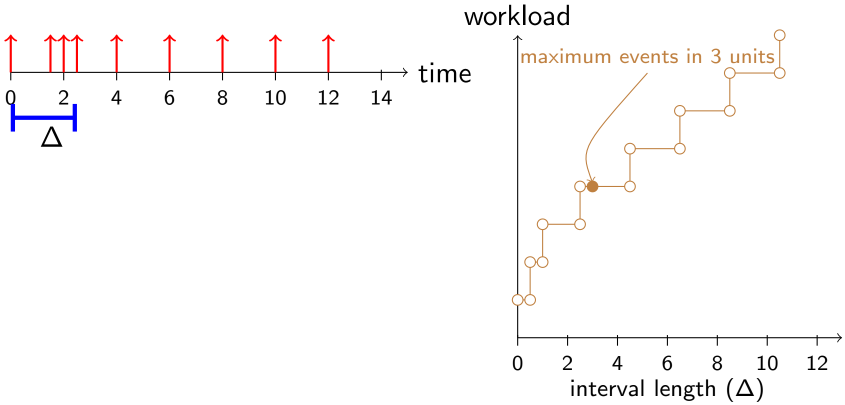
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



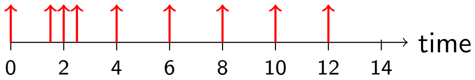
Arrival Curve: An Example

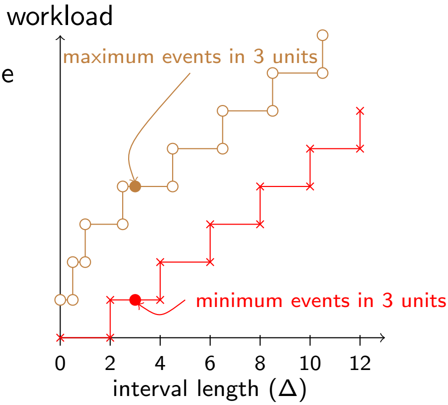
Use a sliding window to get the upper bound of the number of events in a specified interval length.



Arrival Curve: An Example

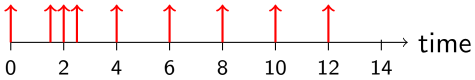
Use a sliding window to get the upper bound of the number of events in a specified interval length.

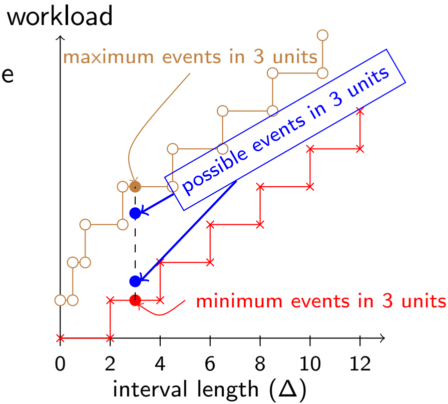




Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.

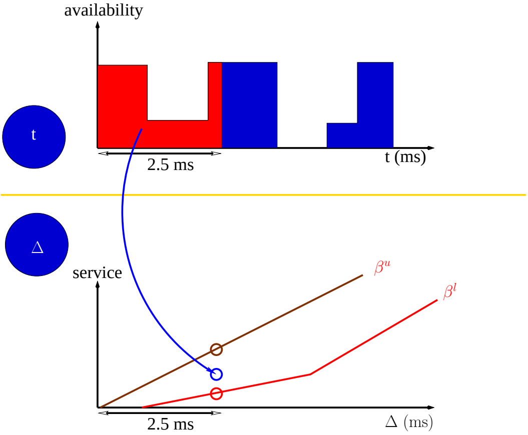




Service Curve: An Example

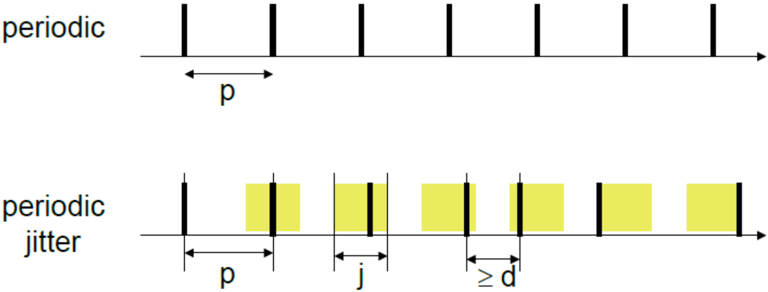
Resource Availability

Service Curves
$\beta = [\beta^l, \beta^u]$



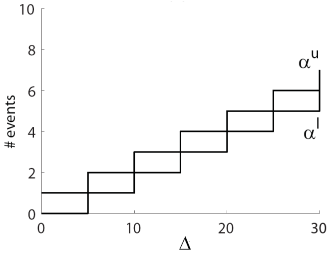
Example 1: Periodic with Jitter

A common event pattern that is used in literature can be specified by the parameter triple (p, j, d), where p denotes the period, j the jitter, and d the minimum inter-arrival distance of events in the modeled stream.



Example 1: Periodic with Jitter

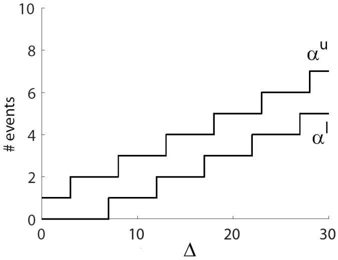
Periodic



$$\alpha^u(\Delta) = \left\lceil \frac{\Delta}{p} \right\rceil$$

$$\alpha^l(\Delta) = \left\lfloor \frac{\Delta}{p} \right\rfloor$$

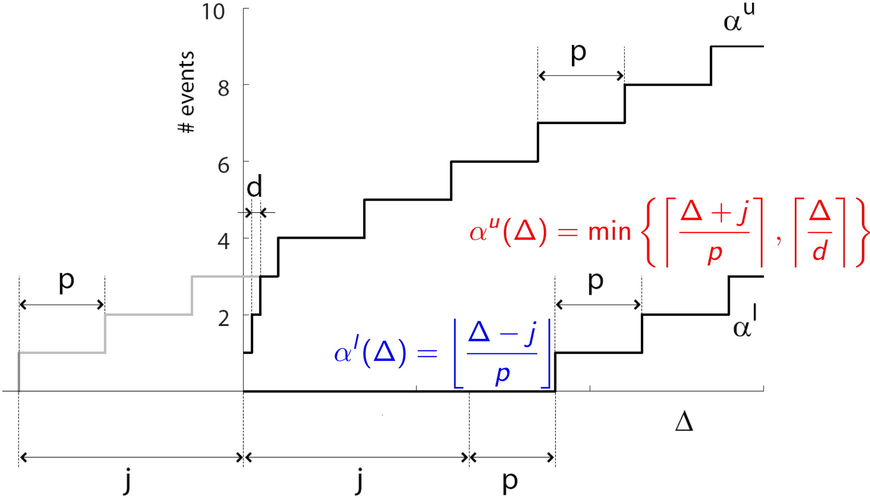
Periodic with Jitter



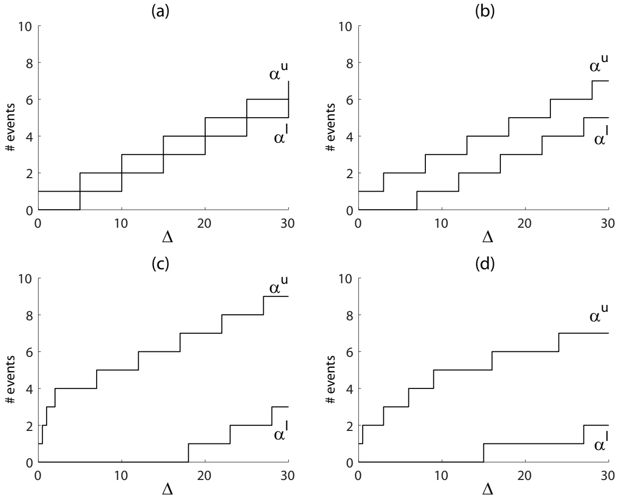
$$\alpha^u(\Delta) = \left\lceil \frac{\Delta + j}{p} \right\rceil$$

$$\alpha^l(\Delta) = \left\lfloor \frac{\Delta - j}{p} \right\rfloor$$

Example 1: Periodic with Jitter

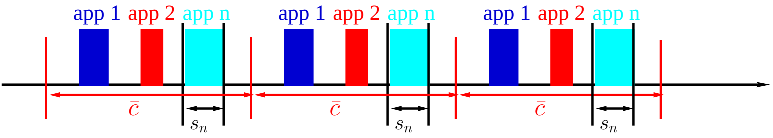


More Examples on Arrival Curves

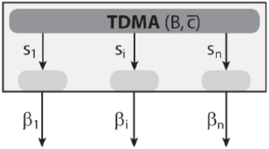


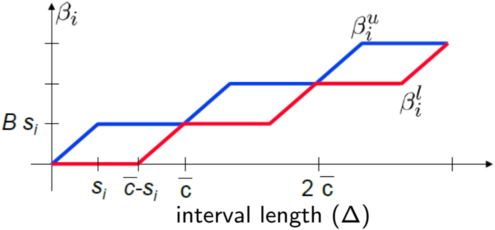
Example 2: TDMA Resource

- Consider a real-time system consisting of n applications that are executed on a resource with bandwidth B that controls resource access using a TDMA (Time Division Multiple Access) policy.
- Analogously, we could consider a distributed system with n communicating nodes, that communicate via a shared bus with bandwidth B, with a bus arbitrator that implements a TDMA policy.
- TDMA policy: In every TDMA cycle of length $\bar{c}$, one single resource slot of length s_i is assigned to application i.



Example 2: TDMA Resource

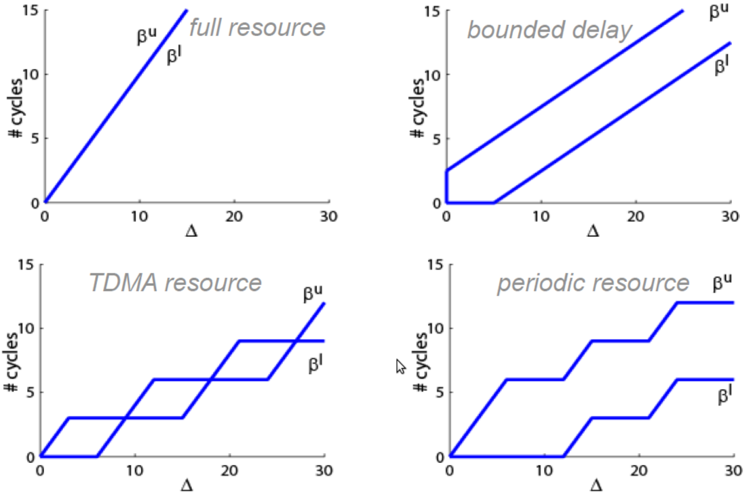




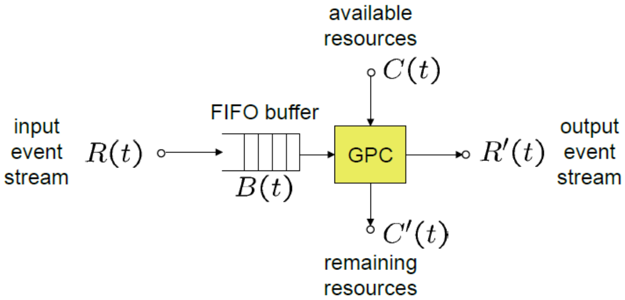
$$\beta^u(\Delta) = B \min\left\{ \left\lceil \frac{\Delta}{\bar{c}} \right\rceil s_i, \Delta - \left\lfloor \frac{\Delta}{\bar{c}} \right\rfloor (\bar{c} - s_i) \right\}$$

$$\beta^l(\Delta) = B \max\left\{ \left\lfloor \frac{\Delta}{\bar{c}} \right\rfloor s_i, \Delta - \left\lceil \frac{\Delta}{\bar{c}} \right\rceil (\bar{c} - s_i) \right\}$$

More Examples on Service Curves



Greedy Processing Component (GPC)



- Component is triggered by incoming events.
- A fully preemptable task is instantiated at every event arrival to process the incoming event.
- Active tasks are processed in a greedy fashion in FIFO order.
- Processing is restricted by the availability of resources.

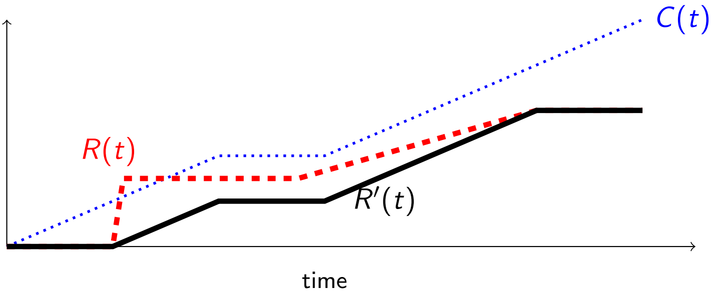
GPC

By conservation law:

$$C(t) = C'(t) + R'(t)$$
$$B(t) = R(t) - R'(t)$$

Therefore,

$$R'(t) = \inf_{0 \le \lambda \le t} \{R(\lambda) + C(t) - C(\lambda)\} \text{ and } C'(t) = \sup_{0 \le \lambda \le t} \{C(\lambda) - R(\lambda)\}$$



Analysis on GPC

- By conservation law: $R'(\lambda) \leq R(\lambda)$ for any $\lambda \geq 0$.
- Since the output cannot be larger than the available resource, we also have $R'(t) \leq R'(\lambda) + C(t) - C(\lambda)$.
- By the two items above, we know $R'(t) \leq R(\lambda) + C(t) - C(\lambda)$.
- Suppose that λ^* is the latest time before t such that the buffer is empty. That is, $R'(\lambda^*) = R(\lambda^*)$ and $R'(t) = R'(\lambda^*) + C(t) - C(\lambda^*) = R(\lambda^*) + C(t) - C(\lambda^*)$.
- As a result, we know that

$$R'(t) = \inf_{0 \leq \lambda \leq t} \{R(\lambda) + C(t) - C(\lambda)\}$$

Analysis on GPC

- By conservation law: $R'(\lambda) \leq R(\lambda)$ for any $\lambda \geq 0$.
- Since the output cannot be larger than the available resource, we also have $R'(t) \leq R'(\lambda) + C(t) - C(\lambda)$.
- By the two items above, we know $R'(t) \leq R(\lambda) + C(t) - C(\lambda)$.
- Suppose that λ^* is the latest time before t such that the buffer is empty. That is, $R'(\lambda^*) = R(\lambda^*)$ and $R'(t) = R'(\lambda^*) + C(t) - C(\lambda^*) = R(\lambda^*) + C(t) - C(\lambda^*)$.
- As a result, we know that

$$R'(t) = \inf_{0 \leq \lambda \leq t} \{R(\lambda) + C(t) - C(\lambda)\}$$

- The analysis is similar for

$$C'(t) = \sup_{0 \leq \lambda \leq t} \{C(\lambda) - R(\lambda)\}$$

Convolutions

- Plus-times system theory: signals f, impulse response g, convolution in time domain:

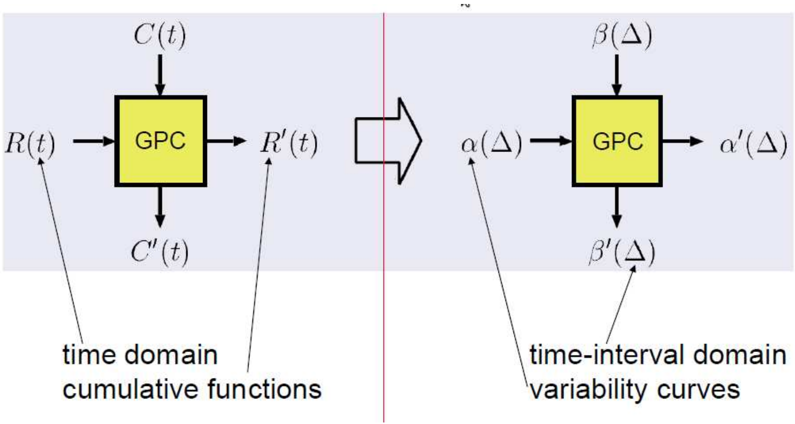
$$h(t) = (f \times g)(t) = \int_0^t f(t-s)g(s)ds,$$

where f, g can be thought of as signals and impulse response, respectively.

- Min-Plus system theory: streams R, variability curves g, convolution in time-interval domain:

$$R'(t) \geq (R \otimes g)(t) = \inf_{0 \leq \lambda \leq t} \{R(t-\lambda) + g(\lambda)\}.$$

Abstraction



Convolution and De-convolution

- $f \otimes g$ is called *min-plus convolution*

$$(f \otimes g)(t) = \inf_{0 \leq \lambda \leq t} \{f(t-\lambda) + g(\lambda)\}$$

- $f \oslash g$ is called *min-plus de-convolution*

$$(f \oslash g)(t) = \sup_{0 \leq \lambda} \{f(t+\lambda) - g(\lambda)\}$$

- $f \bar{\otimes} g$ is called *max-plus convolution*

$$(f \bar{\otimes} g)(t) = \sup_{0 \leq \lambda \leq t} \{f(t-\lambda) + g(\lambda)\}$$

- $f \bar{\oslash} g$ is called *max-plus de-convolution*

$$(f \bar{\oslash} g)(t) = \inf_{0 \leq \lambda} \{f(t+\lambda) - g(\lambda)\}$$

Arrival and Service Curves Revisit

$$\alpha^l(t-s) \leq R(t) - R(s) \leq \alpha^u(t-s) \quad \forall s \leq t.$$

$$\beta^l(t-s) \leq C(t) - C(s) \leq \beta^u(t-s) \quad \forall s \leq t.$$

Therefore, by using the convolution and de-convolution, we know that

$$\alpha^u = R \oslash R; \qquad \alpha^l = R \bar{\oslash} R; \qquad \beta^u = C \oslash C; \qquad \beta^l = C \bar{\oslash} C;$$

The proof for α^u:

$$\alpha^u(\Delta) = \sup_{\lambda \geq 0} \{R(\Delta + \lambda) - R(\lambda)\} \geq R(\Delta + \lambda) - R(\lambda), \;\; \forall \lambda \geq 0.$$

Tight Curves

A curve f is *sub-additive*, if

$$f(a) + f(b) \geq f(a+b) \quad \forall a, b \geq 0.$$

The sub-additive *closure* $\overline{f}$ of a curve f is the largest sub-additive curve with $\overline{f} \leq f$ and is computed as

$$\overline{f} = \min\{f, (f \otimes f), (f \otimes f \otimes f), \ldots\}.$$

If f is interpreted as an arrival curve, then any trace R that is upper bounded by f is also upper bounded by the sub-additive closure $\overline{f}$.

Tight Curves

A curve f is *sub-additive*, if

$$f(a) + f(b) \geq f(a+b) \quad \forall a, b \geq 0.$$

The sub-additive *closure* $\overline{f}$ of a curve f is the largest sub-additive curve with $\overline{f} \leq f$ and is computed as

$$\overline{f} = \min\{f, (f \otimes f), (f \otimes f \otimes f), \ldots\}.$$

If f is interpreted as an arrival curve, then any trace R that is upper bounded by f is also upper bounded by the sub-additive closure $\overline{f}$.

A tight upper arrival curve should satisfy the sub-additive property.

Some Relations

- The output stream of a component satisfies:

$$R'(t) \geq (R \otimes \beta^l)(t)$$

- The output upper arrival curve of a component satisfies

$$\alpha^{u\prime} \leq (\alpha^u \oslash \beta^l)$$

  with a simple and pessimistic calculation.
- The remaining lower service curve of a component satisfies

$$\beta^{l\prime}(\Delta) = \sup_{0 \leq \lambda \leq \Delta} (\beta^l(\lambda) - \alpha^u(\lambda))$$

Some Relations

- The output stream of a component satisfies:

$$R'(t) \geq (R \otimes \beta^l)(t)$$

Proof:
$$R'(t) = \inf_{0 \leq \lambda \leq t} \{R(\lambda) + C(t) - C(\lambda)\}$$
$$\geq \inf_{0 \leq \lambda \leq t} \left\{R(\lambda) + \beta^l(t - \lambda)\right\} = (R \otimes \beta^l)(t).$$

- The output upper arrival curve of a component satisfies

$$\alpha^{u\prime} \leq (\alpha^u \oslash \beta^l)$$

with a simple and pessimistic calculation.

- The remaining lower service curve of a component satisfies

$$\beta^{l\prime}(\Delta) = \sup_{0 \leq \lambda \leq \Delta} (\beta^l(\lambda) - \alpha^u(\lambda))$$

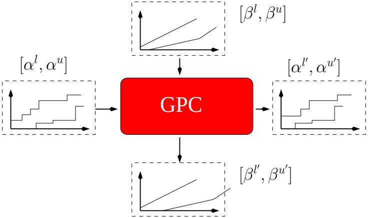
Remaining Service Curve

$$\beta^{l\prime}(\Delta) = \sup_{0 \le \lambda \le \Delta} (\beta^l(\lambda) - \alpha^u(\lambda))$$

$$\begin{aligned}
C'(t) - C'(s) &= \sup_{0 \le a \le t} \{C(a) - R(a)\} - \sup_{0 \le b \le s} \{C(b) - R(b)\} \\
&= \inf_{0 \le b \le s} \left\{ \sup_{0 \le a \le t} \{C(a) - C(b) - (R(a) - R(b))\} \right\} \\
&= \inf_{0 \le b \le s} \left\{ \sup_{0 \le a-b \le t-b} \{C(a) - C(b) - (R(a) - R(b))\} \right\} \\
&\ge \inf_{0 \le b \le s} \left\{ \sup_{0 \le \lambda \le t-b} \left\{ \beta^l(\lambda) - \alpha^u(\lambda) \right\} \right\} \\
&\ge \sup_{0 \le \lambda \le t-s} \left\{ \beta^l(\lambda) - \alpha^u(\lambda) \right\} = \sup_{0 \le \lambda \le \Delta} (\beta^l(\lambda) - \alpha^u(\lambda)).
\end{aligned}$$

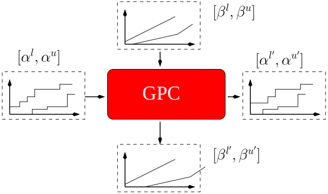
Tighter Bounds

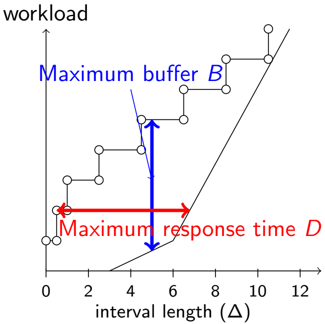
$$\alpha^{u'} = [(\alpha^u \otimes \beta^u) \oslash \beta^l] \wedge \beta^u$$
$$\alpha^{l'} = [(\alpha^u \oslash \beta^l) \otimes \beta^l] \wedge \beta^l$$
$$\beta^{u'} = (\beta^u - \alpha^l)\bar{\oslash}0$$
$$\beta^{l'} = (\beta^l - \alpha^u)\bar{\otimes}0$$



Without formal proofs....

Graphical Interpretation



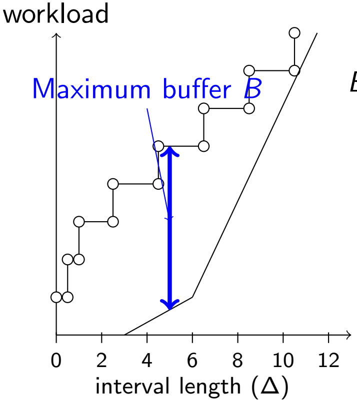


$$B = \sup_{t \geq 0} \{R(t) - R'(t)\} \leq \sup_{\lambda \geq 0} \{\alpha^u(\lambda) - \beta^l(\lambda)\}$$

$$D = \sup_{t \geq 0} \{\inf\{\tau \geq 0 : R(t) \leq R'(t+\tau)\}\}$$

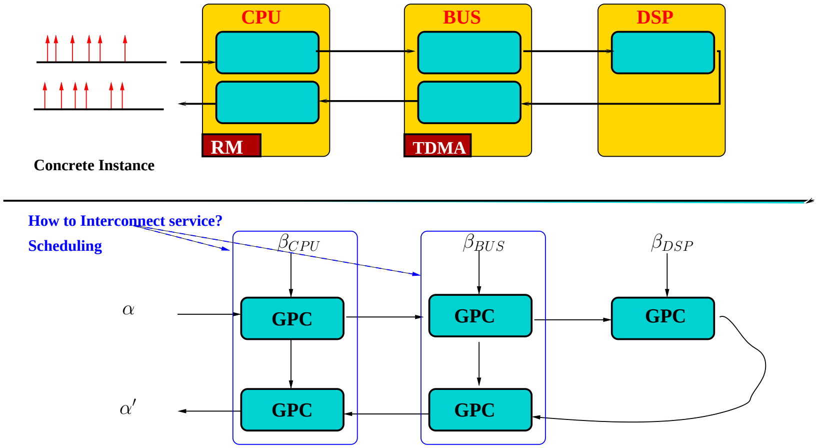
$$= \sup_{\Delta \geq 0} \{\inf\{\tau \geq 0 : \alpha^u(\Delta) \leq \beta^l(\Delta + \tau)\}\}$$

Proof of Buffer Size

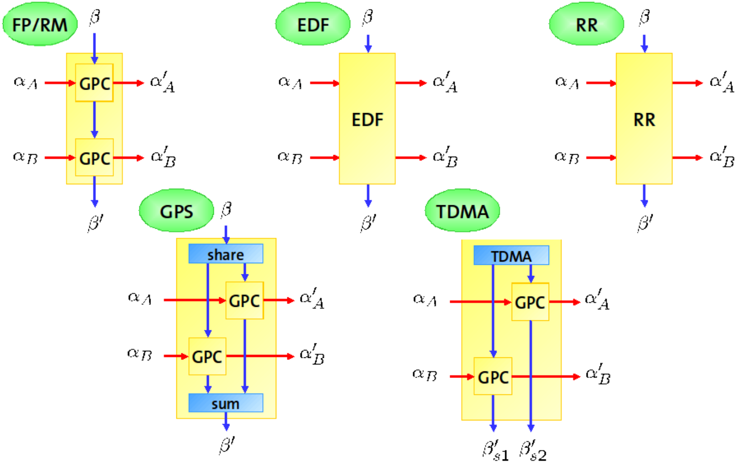


$$
\begin{aligned}
B(t) &= R(t) - R'(t) \\
&= R(t) - \inf_{0 \le u \le t} \{R(u) + C(t) - C(u)\} \\
&= \sup_{0 \le u \le t} \{R(t) - R(u) - C(t) + C(u)\} \\
&= \sup_{0 \le u \le t} \{R(t) - R(u) - (C(t) - C(u))\} \\
&\le \sup_{0 \le u \le t} \{\alpha^u(t-u) - \beta^l(t-u)\} \\
&\le \sup_{0 \le \lambda} \{\alpha^u(\lambda) - \beta^l(\lambda)\}
\end{aligned}
$$

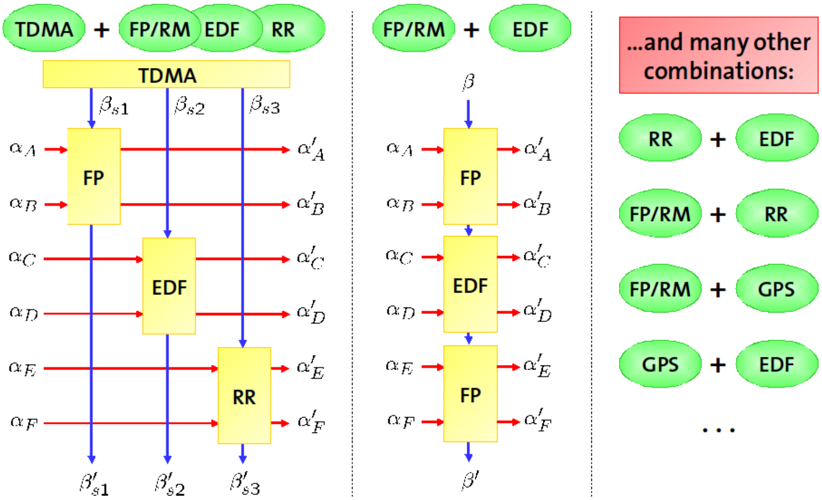
System Composition



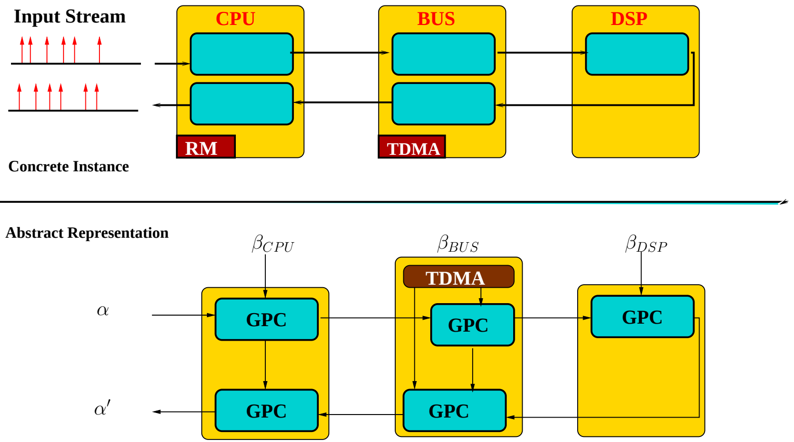
Scheduling and Arbitration



Mixed Hierarchical Scheduling

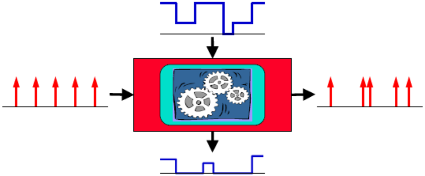


Complete System Composition



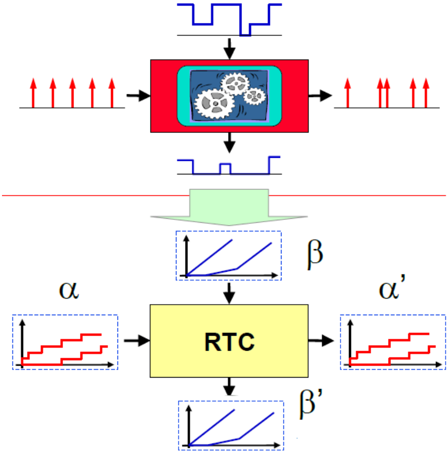
Extending the Framework

- New HW behavior
- New SW behavior
- New scheduling schemes
- New ·········



Extending the Framework

- New HW behavior
- New SW behavior
- New scheduling schemes
- New $\cdots\cdots\cdots$

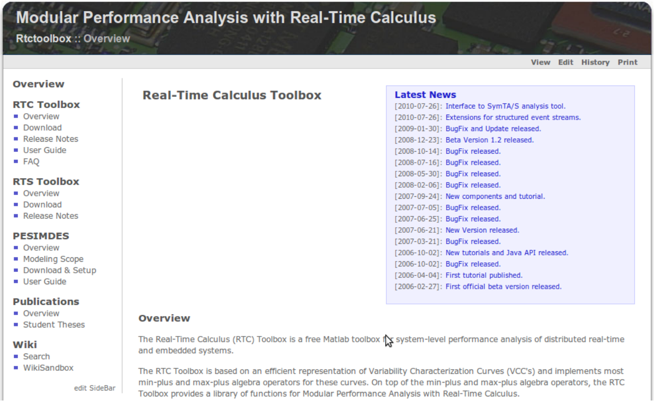


The hard part...

Find new relations

$$\alpha'(\Delta) = f_\alpha(\alpha, \beta)$$
$$\beta'(\Delta) = f_\beta(\alpha, \beta)$$

RTC Toolbox (http://www.mpa.ethz.ch/Rtctoolbox)



Modular Performance Analysis with Real-Time Calculus
Rtctoolbox :: Overview

View Edit History Print

Overview

RTC Toolbox
- Overview
- Download
- Release Notes
- User Guide
- FAQ

RTS Toolbox
- Overview
- Download
- Release Notes

PESIMDES
- Overview
- Modeling Scope
- Download & Setup
- User Guide

Publications
- Overview
- Student Theses

Wiki
- Search
- WikiSandbox

edit SideBar

Real-Time Calculus Toolbox

Latest News
[2010-07-26]: Interface to SymTA/S analysis tool.
[2010-07-26]: Extensions for structured event streams.
[2009-01-30]: BugFix and Update released.
[2008-12-23]: Beta Version 1.2 released.
[2008-10-14]: BugFix released.
[2008-07-16]: BugFix released.
[2008-05-30]: BugFix released.
[2008-02-06]: BugFix released.
[2007-09-24]: New components and tutorial.
[2007-07-05]: BugFix released.
[2007-06-25]: BugFix released.
[2007-06-21]: New Version released.
[2007-03-21]: BugFix released.
[2006-10-02]: New tutorials and Java API released.
[2006-10-02]: BugFix released.
[2006-04-04]: First tutorial published.
[2006-02-27]: First official beta version released.

Overview

The Real-Time Calculus (RTC) Toolbox is a free Matlab toolbox for system-level performance analysis of distributed real-time and embedded systems.

The RTC Toolbox is based on an efficient representation of Variability Characterization Curves (VCC's) and implements most min-plus and max-plus algebra operators for these curves. On top of the min-plus and max-plus algebra operators, the RTC Toolbox provides a library of functions for Modular Performance Analysis with Real-Time Calculus.

Advantages and Disadvantages of RTC and MPA

- Advantages
  - More powerful abstraction than "classical" real-time analysis
  - Resources are first-class citizens of the method
  - Allows composition in terms of (a) tasks, (b) streams, (c) resources, (d) sharing strategies.
- Disadvantages
  - Needs some effort to understand and implement
  - Extension to new arbitration schemes not always simple
  - *Not applicable for schedulers that change the scheduling policies dynamically.*