Hardware-Software Codesign

10. Performance Analysis of Distributed Embedded Systems

Lothar Thiele
System Design

- Specification
- System Synthesis
- SW-Compilation
  - Intellectual Prop. Code
  - Machine Code
- Instruction Set
- HW-Synthesis
  - Intellectual Prop. Block
  - Net lists
- Estimation
Contents

- Overview
- Real-Time Calculus
- Modular Performance Analysis
- Examples
Formal Analysis vs. Simulation

- Worst-Case
- Best-Case

Real System

Simulation

Formal analysis

e.g. delay

upper bound

lower bound
Analysis and Design

Embedded System =
  Computation + Communication + Resource Interaction

Analysis:
Infer system properties from subsystem properties.

Design:
Build a system from subsystems while meeting requirements.
Modular Performance Analysis

Application

Task graphs

Architecture

Architecture diagrams

Service Model (Resources)

Data sheets

Measurements

Load Model (Environment)

Formal specification

Input traces

System Model

Mapping Scheduling

Performance Model

Analysis

Analysis Results

Processing Model (Tasks & Scheduling)

WCET Analysis

Formal specification
Abstract Models for Performance Analysis

Concrete Instance

Abstract Representation

Input Stream

Processor

Task

Concrete Instance

Abstract Representation

Load Model

Service Model

Processing Model
Overview

- Modular Performance Analysis (MPA)
- Real-Time Calculus (RTC)
- Min-Plus Calculus, Max-Plus Calculus
Contents

- Overview
- Real-Time Calculus
- Modular Performance Analysis
- Examples
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:**
Comparison of Algebraic Structures

- **Algebraic structure**
  - set of elements $S$
  - one or more operators defined on elements of this set

- Algebraic structures *with two operators* $\oplus$, $\odot$
  - plus-times: $(S, \oplus, \odot) = (\mathbb{R}, +, \times)$
  - min-plus: $(S, \oplus, \odot) = (\mathbb{R} \cup \{+\infty\}, \inf, +)$

- **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.
  - $\inf\{[3, 4]\} = 3, \quad \inf\{(3, 4]\} = 3$
  - $\min\{[3, 4]\} = 3, \quad \min\{(3, 4]\}$ not defined
Comparison of Algebraic Structures

**Joint properties**

- Closure of $\square$: $a \square b \in S$
- Associativity of $\square$: $a \square (b \square c) = (a \square b) \square c$
- Commutativity of $\square$: $a \square b = b \square a$
- Existence of identity element for $\square$: $\exists \nu : a \square \nu = a$
- Existence of negative element for $\square$: $\exists a^{-1} : a \square a^{-1} = \nu$
- Identity element of $\boxplus$ absorbing for $\square$: $a \square \varepsilon = \varepsilon$
- Distributivity of $\square$ w.r.t. $\boxplus$: $a \square (b \boxplus c) = (a \square b) \boxplus (a \square c)$

**Example:**

- **plus-times:** $a \times (b + c) = a \times b + a \times c$
- **min-plus:** $a + \inf\{b, c\} = \inf\{a + b, a + c\}$
Comparison of Algebraic Structures

- **Joint properties**: □

  - Closure of □: $a \boxplus b \in S$
  - Associativity of □: $a \boxplus (b \boxplus c) = (a \boxplus b) \boxplus c$
  - Commutativity of □: $a \boxplus b = b \boxplus a$
  - Existence of identity element for □: $\exists \varepsilon : a \boxplus \varepsilon = a$

- **Differences** □:
  - *plus-times*: Existence of a negative element for □: $\exists(-a) : a \boxplus (-a) = \varepsilon$
  - *min-plus*: Idempotency of □: $a \boxplus a = a$
Comparison of System Theories

- **Plus-times system theory**
  - signals, impulse response, convolution, time-domain

  \[ h(t) = (f \ast g)(t) = \int_0^t f(t - s) \cdot g(s) \, ds \]

- **Min-plus system theory**
  - streams, variability curves, time-interval domain, convolution

  \[ R(t) \rightarrow g(\Delta) \rightarrow R'(t) \geq (R \otimes g)(t) = \inf_{0 \leq \lambda \leq t} \{ R(t - \lambda) + g(\lambda) \} \]
Abstract Models for Performance Analysis

Concrete Instance

Abstract Representation

Input Stream $R(t)$

Processor $C(t)$

Task

Service Model

Processing Model

Load Model

$\alpha(\Delta)$

$\beta(\Delta)$
From Streams to Cumulative Functions

- **Data streams**: \( R(t) = \) number of events in \([0, t)\)
- **Resource stream**: \( C(t) = \) available resource in \([0, t)\)

![Graph showing R(t) and C(t)]
From Event Streams to Arrival Curves

Event Stream

number of events in in $t=[0 .. 2.5]$ ms

Arrival Curves $\alpha = [\alpha^l, \alpha^u]$

maximum / minimum arriving events in any interval of length 2.5 ms
From Resources to Service Curves

Resource Availability

available service in $t = [0 .. 2.5]$ ms

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

maximum/minimum available service in any interval of length 2.5 ms
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

periodic with jitter
Example 1: Periodic with Jitter

Arrival curves:

\[ \alpha^u(\Delta) = \left\lfloor \frac{\Delta - j}{p} \right\rfloor \]

\[ \alpha^l(\Delta) = \min\left\{ \left\lfloor \frac{\Delta + j}{p} \right\rfloor, \left\lfloor \frac{\Delta}{d} \right\rfloor \right\} \]
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 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

**Service curves** available to the applications / node $i$:

\[
\beta^l_i(\Delta) = B \max\left\{ \frac{\Delta}{\bar{c}} s_i, \Delta - \frac{\Delta}{\bar{c}} (\bar{c} - s_i) \right\}
\]

\[
\beta^u_i(\Delta) = B \min\left\{ \frac{\Delta}{\bar{c}} s_i, \Delta - \frac{\Delta}{\bar{c}} (\bar{c} - s_i) \right\}
\]
Greedy Processing Component (GPC)

**Examples:**
- computation (event – task instance, resource – computing resource [tasks/second])
- communication (event – data packet, resource – bandwidth [packets/second])
Greedy Processing Component

Behavioral Description

- 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.
Greedy Processing Component (GPC)

If the resource and event streams describe available and requested units of processing or communication, then

\[
\begin{align*}
C(t) &= C'(t) + R'(t) \\
B(t) &= R(t) - R'(t)
\end{align*}
\]

Conservation Laws

\[
R'(t) = \inf_{0 \leq u \leq t} \{ R(u) + C(t) - C(u) \} 
\]
Greedy Processing

- For all times $u \leq t$ we have $R'(u) \leq R(u)$ (conservation law).
- We also have $R'(t) \leq R'(u) + C(t) - C(u)$ as the output can not be larger than the available resources.
- Combining both statements yields $R'(t) \leq R(u) + C(t) - C(u)$.
- Let us suppose that $u^*$ is the last time before $t$ with an empty buffer. We have $R(u^*) = R'(u^*)$ at $u^*$ and also $R'(t) = R'(u^*) + C(t) - C(u^*)$ as all available resources are used to produce output. Therefore, $R'(t) = R(u^*) + C(t) - C(u^*)$.
- As a result, we obtain

$$R'(t) = \inf_{0 \leq u \leq t} \{ R(u) + C(t) - C(u) \}$$
Abstract Models for Performance Analysis

Concrete Instance

Abstract Representation

Input Stream

Processor C(t)

Task

Service Model

Load Model

Processing Model

\[ R(t) \]

\[ \alpha(\Delta) \]

\[ \beta(\Delta) \]
Abstraction

$$R(t) \xrightarrow{GPC} R'(t)$$

$$C(t)$$

$$C''(t)$$

time domain cumulative functions

time-interval domain variability curves

$$\alpha(\Delta) \xrightarrow{GPC} \alpha'(\Delta)$$

$$\beta(\Delta)$$

$$\beta'(\Delta)$$
Some Definitions and Relations

- $f \otimes g$ is called **min-plus convolution**
  \[(f \otimes g)(t) = \inf_{0 \leq u \leq t} \{ f(t - u) + g(u) \}\]

- $f \oslash g$ is called **min-plus de-convolution**
  \[(f \oslash g)(t) = \sup_{u \geq 0} \{ f(t + u) - g(u) \}\]

- For **max-plus convolution and de-convolution**:
  \[(f \bar{\otimes} g)(t) = \sup_{0 \leq u \leq t} \{ f(t - u) + g(u) \}\]
  \[(f \bar{\oslash} g)(t) = \inf_{u \geq 0} \{ f(t + u) - g(u) \}\]

- Relation between convolution and deconvolution
  \[f \leq g \otimes h \iff f \oslash h \leq g\]
Arrival and Service Curve

- The arrival and service curves provide bounds on event and resource functions as follows:
  \[
  \alpha^l(t - s) \leq R(t) - R(s) \leq \alpha^u(t - s) \quad \forall s \leq t
  \]
  \[
  \beta^l(t - s) \leq C(t) - C(s) \leq \beta^u(t - s) \quad \forall s \leq t
  \]

- We can determine valid variability curves from cumulative functions as follows:
  \[
  \alpha^u = R \ominus R; \quad \alpha^l = R \ominus R; \quad \beta^u = C \ominus C; \quad \beta^l = C \ominus C
  \]

- One proof:
  \[
  \alpha^u = R \ominus R \Rightarrow \alpha^u(\Delta) = \sup_{u \geq 0} \{R(\Delta + u) - R(u)\} \Rightarrow
  \]
  \[
  \alpha^u(\Delta) = \sup_{s \geq 0} \{R(\Delta + s) - R(s)\} \Rightarrow \alpha^u(t-s) \geq R(t) - R(s) \quad \forall t \geq s
  \]
Abstraction

\[ R(t) \rightarrow GPC \rightarrow R'(t) \]

\[ C(t) \rightarrow GPC \rightarrow C''(t) \]

\[ \alpha(\Delta) \rightarrow GPC \rightarrow \alpha'(\Delta) \]

\[ \beta(\Delta) \rightarrow GPC \rightarrow \beta'(\Delta) \]

- Time domain cumulative functions
- Time-interval domain variability curves

Swiss Federal Institute of Technology
The Most Simple 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:
  \[ \alpha'^u = (\alpha^u \otimes \beta^l) \]

- The remaining lower service curve of a component satisfies:
  \[ \beta'^l(\Delta) = \sup_{0 \leq \lambda \leq \Delta} (\beta^l(\lambda) - \alpha^u(\lambda)) \]
Two Sample Proofs

\[ R'(t) \geq (R \otimes \beta^l)(t) \]

\[
R'(t) = \inf_{0 \leq u \leq t} \{ R(u) + C(t) - C(u) \} \\
\geq \inf_{0 \leq u \leq t} \{ R(u) + \beta^l(t - u) \} \\
= (R \otimes \beta^l)(t)
\]

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

\[
C'(t) - C'(s) = \sup_{0 \leq a \leq t} \{ C'(a) - R(a) \} - \sup_{0 \leq b \leq s} \{ C'(b) - R(b) \} = \\
= \inf_{0 \leq b \leq s} \{ \sup_{0 \leq a \leq t} \{ (C'(a) - C'(b)) - (R(a) - R(b)) \} \} \\
= \inf_{0 \leq b \leq s} \{ \sup_{0 \leq a - b \leq t - b} \{ (C'(a) - C'(b)) - (R(a) - R(b)) \} \} \\
\geq \inf_{0 \leq b \leq s} \{ \sup_{0 \leq \lambda \leq t - b} \{ \beta^l(\lambda) - \alpha^u(\lambda) \} \} \geq \sup_{0 \leq \lambda \leq t - s} \{ \beta^l(\lambda) - \alpha^u(\lambda) \}
\]
Tighter Bounds

The greedy processing component transforms the variability curves as follows:

\[ \alpha^{u'} = [(\alpha^u \otimes \beta^u) \otimes \beta^l] \land \beta^u \]
\[ \alpha^l' = [(\alpha^l \otimes \beta^u) \otimes \beta^l] \land \beta^l \]
\[ \beta^{u'} = (\beta^u - \alpha^l) \ominus 0 \]
\[ \beta^l' = (\beta^l - \alpha^u) \ominus 0 \]

Without proof ... .
Delay and Backlog

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

\[ D = \sup_{t \geq 0} \{ \inf \{ \tau \geq 0 : R(t) \leq R'(t + \tau) \} \} \]
\[ = \sup_{\Delta \geq 0} \{ \inf \{ \tau \geq 0 : \alpha^u(\Delta) \leq \beta^l(\Delta + \tau) \} \} \]
Proof of Backlog Bound

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

\[ B(t) = R(t) - R'(t) = R(t) - \inf_{0 \leq u \leq t} \{R(u) + C(t) - C(u)\} \]

\[ = \sup_{0 \leq u \leq t} \{(R(t) - R(u)) - (C(t) - C(u))\} \]

\[ \leq \sup_{0 \leq u \leq t} \{\alpha^u(t - u) - \beta^l(t - u)\} \]

\[ \leq \sup_{0 \leq \lambda} \{\alpha^u(\lambda) - \beta^l(\lambda)\} \]
Contents

- Overview
- Real-Time Calculus
- Modular Performance Analysis
- Examples
System Composition

How to interconnect service?

Scheduling!

\[ \beta_{CPU} \]

\[ \beta_{BUS} \]

\[ \beta_{DSP} \]

\[ \alpha \]

\[ \alpha' \]
Scheduling and Arbitration

FP/RM  β

GPC  α_A

α'_A

GPC  α_B

α'_B

GPS  β

β'

EDF

GPC  α_A

α'_A

GPC  α_B

α'_B

EDF  β

β'

RR

GPC  α_A

α'_A

GPC  α_B

α'_B

RR  β

β'

TDMA

share

GPC  α_A

α'_A

GPC  α_B

α'_B

TDMA

GPC  α_A

α'_A

GPC  α_B

α'_B

β'
Complete System Composition

**Diagram: Complete System Composition**

- **CPU**
  - RM
  - TDMA

- **BUS**
  - RM
  - TDMA

- **DSP**
  - GPC

**Equations:**

\[ \beta_{CPU} \]
\[ \beta_{BUS} \]
\[ \beta_{DSP} \]

**Variables:**

\[ \alpha, \alpha' \]
Extending the Framework

- New HW behavior
- New SW behavior
- New scheduling scheme
- ...

Find new relations:

\[
\alpha'(\Delta) = f_\alpha(\alpha, \beta) \\
\beta'(\Delta) = f_\beta(\alpha, \beta)
\]

This is the hard part…!
Contents

» Overview

» Real-Time Calculus

» Modular Performance Analysis

» Examples
Case Study

6 Real-Time Input Streams
- with jitter
- with bursts
- deadline > period

3 ECU’s with own CC’s

13 Tasks & 7 Messages
- with different WCET

2 Scheduling Policies
- Earliest Deadline First (ECU’s)
- Fixed Priority (ECU’s & CC’s)

Hierarchical Scheduling
- Static & Dynamic Polling Servers

Bus with TDMA
- 4 time slots with different lengths
  (#1,#3 for CC1, #2 for CC3, #4 for CC3)

Total Utilization:
- ECU1 59 %
- ECU2 87 %
- ECU3 67 %
- BUS 56 %
## Specification Data

<table>
<thead>
<tr>
<th>Stream</th>
<th>(p,j,d) [ms]</th>
<th>D [s]</th>
<th>Task Chain</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>(1000, 2000, 25)</td>
<td>8.0</td>
<td>T1.1 → C1.1 → T1.2 → C1.2 → T1.3</td>
</tr>
<tr>
<td>S2</td>
<td>(400, 1500, 50)</td>
<td>1.8</td>
<td>T2.1 → C2.1 → T2.2</td>
</tr>
<tr>
<td>S3</td>
<td>(600, 0, -)</td>
<td>6.0</td>
<td>T3.1 → C3.1 → T3.2 → C3.2 → T3.3</td>
</tr>
<tr>
<td>S4</td>
<td>(20, 5, -)</td>
<td>0.5</td>
<td>T4.1 → C4.1 → T4.2</td>
</tr>
<tr>
<td>S5</td>
<td>(30, 0, -)</td>
<td>0.7</td>
<td>T4.1 → C4.1 → T4.2</td>
</tr>
<tr>
<td>S6</td>
<td>(1500, 4000, 100)</td>
<td>3.0</td>
<td>T6.1</td>
</tr>
</tbody>
</table>

### Task e

<table>
<thead>
<tr>
<th>Task</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1.1</td>
<td>200</td>
</tr>
<tr>
<td>T1.2</td>
<td>300</td>
</tr>
<tr>
<td>T1.3</td>
<td>30</td>
</tr>
<tr>
<td>T2.1</td>
<td>75</td>
</tr>
<tr>
<td>T2.2</td>
<td>25</td>
</tr>
<tr>
<td>T3.1</td>
<td>60</td>
</tr>
<tr>
<td>T3.2</td>
<td>60</td>
</tr>
<tr>
<td>T3.3</td>
<td>40</td>
</tr>
<tr>
<td>T4.1</td>
<td>12</td>
</tr>
<tr>
<td>T4.2</td>
<td>2</td>
</tr>
<tr>
<td>T5.1</td>
<td>8</td>
</tr>
<tr>
<td>T5.2</td>
<td>3</td>
</tr>
<tr>
<td>T6.1</td>
<td>100</td>
</tr>
</tbody>
</table>

### Message e

<table>
<thead>
<tr>
<th>Message</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1.1</td>
<td>100</td>
</tr>
<tr>
<td>C1.2</td>
<td>80</td>
</tr>
<tr>
<td>C2.1</td>
<td>40</td>
</tr>
<tr>
<td>C3.1</td>
<td>25</td>
</tr>
<tr>
<td>C3.2</td>
<td>10</td>
</tr>
<tr>
<td>C4.1</td>
<td>3</td>
</tr>
<tr>
<td>C5.1</td>
<td>2</td>
</tr>
</tbody>
</table>

### Periodic Server

<table>
<thead>
<tr>
<th>Peridodic Server</th>
<th>p</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPS_{ECU1}</td>
<td>500</td>
<td>200</td>
</tr>
<tr>
<td>SPS_{ECU3}</td>
<td>500</td>
<td>250</td>
</tr>
<tr>
<td>DPS_{ECU3}</td>
<td>600</td>
<td>120</td>
</tr>
</tbody>
</table>

### TDMA t

<table>
<thead>
<tr>
<th>TDMA</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cycle</td>
<td>100</td>
</tr>
<tr>
<td>Slot_{CC1a}</td>
<td>20</td>
</tr>
<tr>
<td>Slot_{CC1b}</td>
<td>25</td>
</tr>
<tr>
<td>Slot_{CC2}</td>
<td>25</td>
</tr>
<tr>
<td>Slot_{CC3}</td>
<td>30</td>
</tr>
</tbody>
</table>
The Distributed Embedded System...
... and its MPA Model
Available & Remaining Service of ECU1
Input of Stream 3
Output of Stream 3
Automated Design Space Exploration

We use evolutionary algorithms for multi-objective optimization!
Network Processor Task Model
Results

Performance for encryption/decryption

Performance for RT voice processing

- **DSP**
  - NRT: 64%
  - RT: 39%

- **Cipher**
  - NRT: 71%
  - RT: 0%

- **LookUp**
  - NRT: 15%
  - RT: 6%

- **Classifier**
  - NRT: 27%
  - RT: 11%

- **DSP**
  - NRT: 35%
  - RT: 39%

- **LookUp**
  - NRT: 1%
  - RT: 6%

- **Classifier**
  - NRT: 1%
  - RT: 11%
Analysis vs. Simulation

![Graphs showing Analysis vs. Simulation](image)

- Simulation
- Analytical Method

Packet Size [Bytes]

Utilization [%]

Load

250 Mbps
200 Mbps
150 Mbps
100 Mbps
Design Space Exploration

- Determine mapping
- Determine performance network
- Solve system of equations
- Determine important parameters (end-to-end delay, throughput, buffer space output jitter, …)
- Give feedback to optimization
RTC Toolbox

Real-Time Calculus Toolbox

Overview

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

www.mpa.ethz.ch/rtctoolbox