Formal Modeling and Analysis of Stream Processing Systems

(cont.)

Linh T.X. Phan

March 2009
Previous Lecture...

- General concepts of the performance analysis and design of stream processing systems

- Simulation vs formal analysis

- Existing formal analysis methods: pros and cons

- **Real-Time Calculus (RTC)**
  - High-level overview
  - Count-based abstraction
  - Definition of arrival and service functions
Real-Time Calculus (cont.)

• A brief introduction to RTC
  – Refer to reading list for more!

• Materials are based on
  – Le Boudec and Thiran’s book on Network Calculus
  – The MPA framework
Recall… Event Streams

• Infinite sequences of data items (events)
• A concrete arrival pattern can be described as a cumulative function $R(t)$
  – $R(t) = \# \text{items arrive in the time interval } [0,t)$
• All possible arrival patterns of an event stream is abstracted as an arrival function $\alpha(\Delta)$
  – $[\alpha^l(\Delta), \alpha^u(\Delta)]$ : the min. and max. number of events that arrive in any time interval of length $\Delta$
An Arrival Pattern

\[ R(t) = \text{number of events that arrive in } [0, t) \]

\#events that arrive in \([t, t+\Delta)\) is: \(R(t+\Delta) - R(t)\)
Arrival Function of A Set of Concrete Patterns

\[ R_1(t) \]
\[ R_2(t) \]
\[ \alpha^u(\Delta) \]
\[ \alpha^l(\Delta) \]
Recall… Resources

• A concrete service pattern
  – how much and when the resource is available
  – captured as a cumulative function \( C(t) \) which gives the amount of resource units available in time interval \([0,t)\)

• All possible service patterns of a resource is abstracted as a service function \( \beta(\Delta) \)
  – \([\beta^l(\Delta), \beta^u(\Delta)]\) : the min. and max. number of resource units available (or the number of events that can be processed) in any time interval of length \(\Delta\)
Service Function of A Set of Concrete Patterns

\[ \beta^u(\Delta) \]

\[ \beta^l(\Delta) \]

\[ C_1(t) \]

\[ C_2(t) \]
Examples of Arrival and Service Functions
Standard Event Streams

Periodic

\[ p \]

Periodic with burst

\[ p \quad j \quad \geq d \]

\[ p: \text{period} \quad j: \text{jitter} \quad d: \text{minimum inter-arrival distance} \]
\[ \alpha^l(\Delta) = \left\lfloor \frac{\Delta}{p} \right\rfloor \quad \alpha^u(\Delta) = \left\lceil \frac{\Delta}{p} \right\rceil \]

periodic arrival function

\begin{align*}
\text{# events} & \quad \Delta \\
0 & \quad p & \quad 2p & \quad 3p & \quad 4p & \quad 5p
\end{align*}
\[ \alpha^l(\Delta) = \left\lfloor \frac{\Delta - j}{p} \right\rfloor \quad \alpha^u(\Delta) = \min \left\{ \left\lceil \frac{\Delta + j}{p} \right\rceil, \left\lceil \frac{\Delta}{d} \right\rceil \right\} \]
Common Resources

The processor is always available

\[ f = \text{processor frequency} \]

**Full resource service function**

\[ \beta^u = \beta^l \]

The diagram shows the relationship between # cycles and \( \Delta \) with the condition that the processor is always available.
Bounded Delay

\[ \forall t, \forall t' \geq t : (t' - t - D)f \leq C(t') - C(t) \leq (t' - t + D)f \]

**bounded delay service function**

\[ \beta^u \]
\[ \beta^l \]

\[ f = \text{frequency} \]
\[ D = \text{bounded delay} \]
TDMA Resource

- A shared resource of bandwidth $B$
- $n$ applications: $App_1$, …, $App_n$
- TDMA policy
  - a resource slot of length $s_i$ is assigned to $App_i$ in every cycle of length $c$
  - the resource given to $App_i$ is bounded by

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

$$
\beta^u_i(\Delta) = B \min \left\{ \left\lfloor \frac{\Delta}{c} \right\rfloor s_i, \Delta - \left\lfloor \frac{\Delta}{c} \right\rfloor (c - s_i) \right\}
$$
TDMA Resource

- Graph showing # cycles vs. Δ with points B.s_i, s_i, c-s_i, c, and lines β^u_i, β^l_i.
The functions $f_\alpha$, $f_\beta$, $Buf$, $Del$ must take into account the scheduling policy and the processing semantics of the component.
Processing Model: Abstract Component

• Relate input arrival/service functions and
  – output arrival and service functions
  – maximum backlog
  – maximum delay

• The computation must capture the way input event streams are processed by the resource

• Vary depending on the scheduling policy and processing semantics, but always deterministic

• Based on min-plus and max-plus algebra
A concrete system component

\[
\begin{align*}
R(t) &\rightarrow C(t) &\rightarrow GPC &\rightarrow R'(t) \\
&\downarrow \quad ? \quad \downarrow \quad ? \quad \downarrow \quad ? \quad \downarrow \\
&\quad C'(t) \\
\end{align*}
\]

- an arrival pattern of the input stream
- a service pattern of the available resource
- an arrival pattern of the output stream
- a service pattern of the remaining resource
Greedy Processing Component

- Triggered by incoming events
  - a preemptive task is instantiated to process each arrival event
- Events are processed in a greedy fashion and FIFO order
  - subjected to resource availability
  - waiting events are stored in the input buffer
- Backlog at time $t$
  - $B(t) = \#\text{events in the buffer at time } t$
- Delay at time $t$
  - $d(t) = \text{the maximum processing time (including waiting time) of an event arriving before } t$
$R(t) \rightarrow \text{GPC} \rightarrow R'(t)$

$C(t) \rightarrow R'(t)$

$C'(t)$

$\# \text{ events}$

$R(t)$

$C(t)$

$R'(t)$

continue to process again

no cycle to process

unused resource

no additional output events

buffer empty here!

$u_0$
GPC: Output Stream

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

For all \( u \leq t \):
- \( R'(u) \leq R(u) \) and \( R'(t) \leq R'(u) + C(t) - C(u) \)
  - #output-events in \([0,u)\) is no more than #input-events in \([0,u)\)
  - #output-events in \([u,t)\) is no more than #events that can be processed in \([u,t)\)

Hence, \( R'(t) \leq R(u) + C(t) - C(u) \)

- Let \( u_0 \) be the last instant before \( t \) at which \( B(u_0) = 0 \)
  - \( R'(u_0) = R(u_0) \); \( R'(t) = R(u_0) + C(t) - C(u_0) \)
  - Thus, \( R'(t) = R'(u_0) + C(t) - C(u_0) \)
Conservative use of resource:

\[ C(t) = C'(t) + R'(t) \]

GPC: Remaining Resource
Backlog at time $t$: $B(t) = R(t) - R'(t)$

Maximum backlog: $B_{max} = \max_{t \geq 0} B(t)$
Delay at time $t$:

$$d(t) = \min\{\lambda : R'(t + \lambda) \geq R(t)\}$$
An abstract system component

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

- The arrival function of the input stream
- The service function of the available resource
- The arrival function of the output stream
- The service pattern of the remaining resource
Basic Min-plus/Max-plus Operators

• Min-plus convolution and de-convolution

\[(f \otimes g)(t) = \inf_{0 \leq u \leq t} \{f(t - u) + g(u)\}\]

\[(f \oslash g)(t) = \sup_{u \geq 0} \{f(t + u) - g(u)\}\]

• Max-plus convolution and de-convolution

\[(f \bar{\otimes} g)(t) = \sup_{0 \leq u \leq t} \{f(t - u) + g(u)\}\]

\[(f \bar{\oslash} g)(t) = \inf_{u \geq 0} \{f(t + u) - g(u)\}\]
e.g. Min-Conv of Convex Piecewise Linear Curves

- Label segments in increasing slope order
- Connect segments end to end with increasing slope
Basic Min-plus/Max-plus Operators

- Recall for all \( t, \Delta \geq 0 \)
  \[
  \alpha^l(\Delta) \leq R(t + \Delta) - R(t) \leq \alpha^u(\Delta)
  \]
  \[
  \beta^l(\Delta) \leq C(t + \Delta) - C(t) \leq \beta^u(\Delta)
  \]
- Valid arrival and service functions for a given \( R(t) \) and \( C(t) \)

\[
\alpha^l = R \ominus R \quad \alpha^u = R \odot R 
\]
\[
\beta^l = C \ominus C \quad \beta^u = C \odot C
\]
GPC: Output Bounds

\[
\alpha^{l'} = \min \left\{ (\alpha^l \ominus \beta^u) \odot \beta^l, \beta^l \right\}
\]

\[
\alpha^{u'} = \min \left\{ (\alpha^u \ominus \beta^u) \odot \beta^l, \beta^u \right\}
\]

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

\[
\beta^{u'} = (\beta^u - \alpha^l) \bar{\otimes} 0
\]
Compute $\alpha^u$ - Intuitive Idea

$$\max\_output(\Delta+\lambda) \leq \max\_input(\Delta+\lambda-\tau) + \max\_processed(\tau), \ \forall \ 0 \leq \tau \leq \Delta+\lambda$$

$$\leq \sup_{0 \leq \tau \leq \Delta+\lambda} \{ \alpha^u(\Delta+\lambda-\tau) + \beta^u(\tau) \} = \gamma(\Delta+\lambda), \text{ with } \gamma = \alpha^u \otimes \beta^u$$

$$\max\_output(\Delta) \leq \max\_output(\Delta+\lambda) - \min\_processed(\lambda), \ \forall \ \lambda \geq 0$$

$$\leq \gamma(\Delta+\lambda) - \beta^l(\lambda), \ \forall \ \lambda \geq 0$$

$$\leq \inf_{\lambda \geq 0} \{ \gamma(\Delta+\lambda) - \beta^l(\lambda) \} = (\gamma \otimes \beta^l)(\Delta)$$

$$\leq [\alpha^u \otimes \beta^u] \otimes \beta^l \otimes (\Delta)$$

Further, $\max\_output(\Delta) \leq \max\_processed(\Delta) \leq \beta^u(\Delta)$

$$\Rightarrow \alpha'^u \leq \min \{ \alpha^u \otimes \beta^u \otimes \beta^l, \beta^u \}$$
GPC: Backlog and Delay Bounds

\[ B_{\text{max}} = \sup_{t \geq 0} \{ R(t) - R'(t) \} \]
\[ \leq \sup_{\Delta \geq 0} \{ \alpha^u(\Delta) - \beta^l(\Delta) \} \]

\[ D_{\text{max}} = \sup_{t \geq 0} \left\{ \inf \{ u \geq 0 : R(t) \leq R'(t + u) \} \right\} \]
\[ = \sup_{\Delta \geq 0} \left\{ \inf \{ u \geq 0 : \alpha^u(\Delta) \leq \beta^l(\Delta + u) \} \right\} \]
GPC: Backlog and Delay Bounds

Maximum vertical distance

Maximum horizontal distance
Scheduling Multiple Event Streams

Fixed Priority:

• video stream has higher priority than audio stream
  ➔ process the video stream first

• remaining resource is used to process the audio stream
Fixed Priority Scheduling

![Diagram of Fixed Priority Scheduling]
TDMA Scheduling

\( b_i \): computed based on the length of the TDMA cycle \( c \) and the slot \( s_i \)

\[
\beta_i^l(\Delta) = B \max \left\{ \left\lfloor \frac{\Delta}{c} \right\rfloor s_i, \Delta - \left\lfloor \frac{\Delta}{c} \right\rfloor (c - s_i) \right\}
\]
\[
\beta_i^u(\Delta) = B \min \left\{ \left\lfloor \frac{\Delta}{c} \right\rfloor s_i, \Delta - \left\lfloor \frac{\Delta}{c} \right\rfloor (c - s_i) \right\}
\]
Modular Performance Analysis using RTC

Diagram showing the flow of video streams through processing elements (PEs) and the IDCT module.
Mixed Hierarchical Scheduling
The RTC Toolbox

www.mpa.ethz.ch/rtctoolbox
The RTC Toolbox

Matlab Command Line | Simulink

RTC Toolbox

www.mpa.ethz.ch/rtctoolbox

Java API

Min-Plus/Max-Plus Algebra, Utilities

Efficient Curve Representation
RTC - Summary

- **Modeling: count-based abstraction**
  - captures burstiness of event streams and variability of the resources as functions

- **Analysis: min-plus and max-plus algebra**
  - can be computed efficiently with tool support

- **Modular and compositional**
  - possible combination with other methods, e.g. standard event models, ECA, simulation

- **Modeling of state-dependencies is difficult**
  - extension of RTC: an active area of study
  - various work combines concepts in RTC with automata
References and Readings

Real-Time Calculus:

1. Jean-Yves Le Boudec and Patrick Thiran: "Network Calculus", Lecture Notes in Computer Science 2050, Springer Verlag, January 2004 (Chapter 1 & 3)


5. Many other papers can be found in the RTC Toolbox website
References and Readings

Other Formal Analysis Methods

1. Kai Richter and Rolf Ernst: "Event Model Interfaces for Heterogeneous System Analysis", Design Automation and Test in Europe Conference (DATE), 2002


5. C. Norstrom, A. Wall and W. Yi: "Timed automata as task models for event-driven systems", 6th International Workshop on Real-Time Computing and Applications Symposium (RTCSA), Hong Kong, China, 1999
References and Readings

Hybrids of RTC and others:


Reference and Readings

Simulation and Trace-based Analysis


linhphan AT seas.upenn.edu

Office: Room 279 South