A General Framework for Analysing System Properties in Platform-Based Embedded System Designs

Samarjit Chakraborty,
Simon Künzli, Lothar Thiele

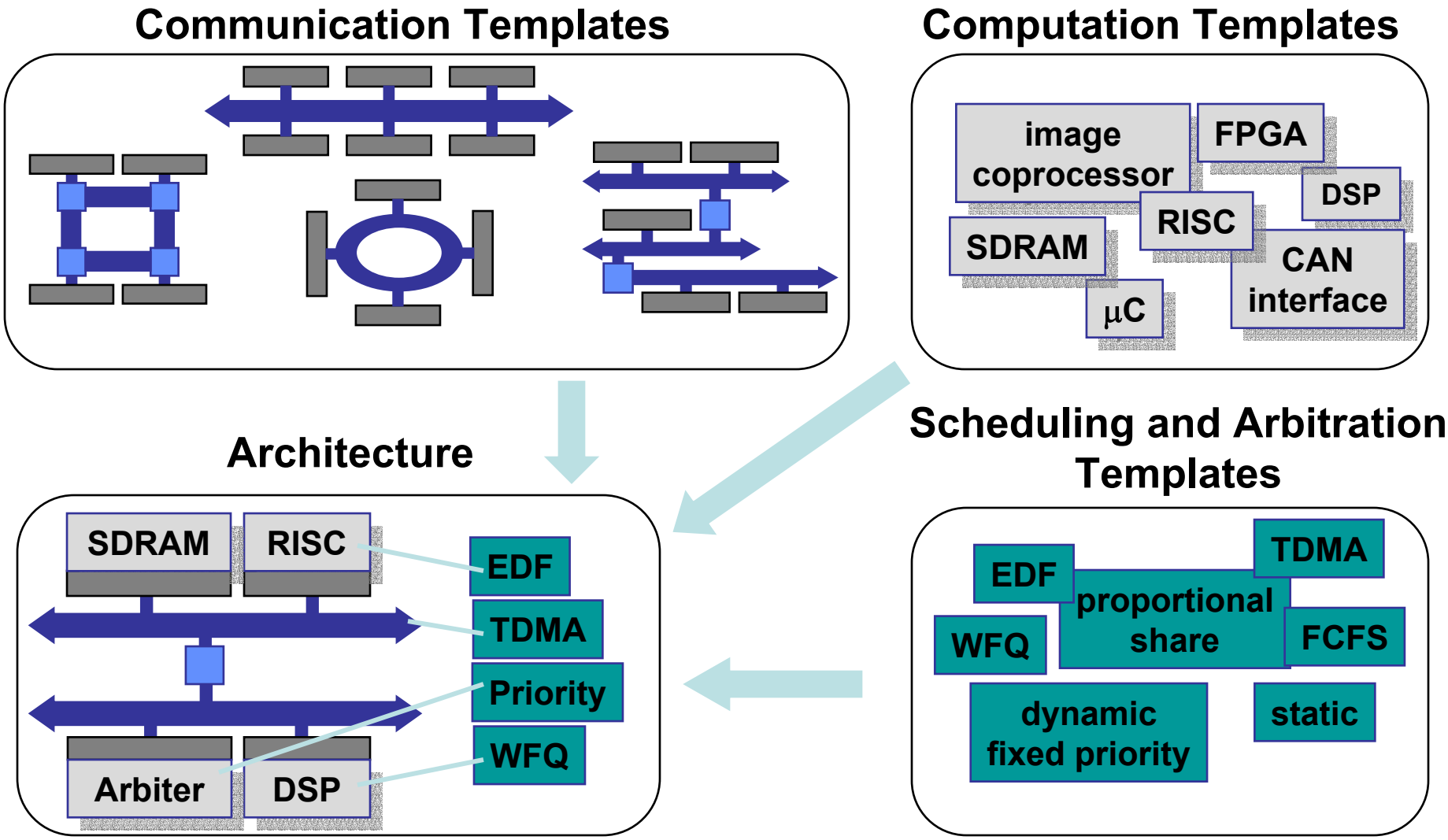
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

TIK
Institut für Technische Informatik
und Kommunikationsnetze
Computer Engineering and Networks Laboratory

Outline

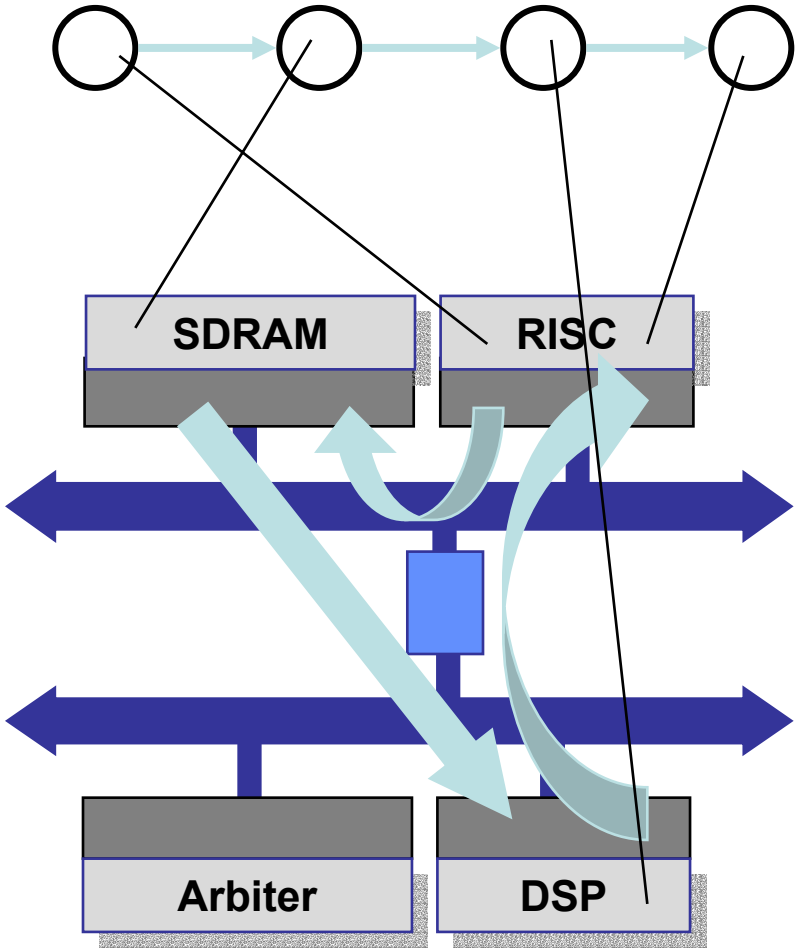
- Introduction
  - Platform-Based Design – Goals and the Reality
- Models
  - Architecture
  - Event Streams, Task Processing, Resources
- Real-Time Calculus
  - Delays, On-Chip Memory, Utilization
  - Scheduling Strategies
- Illustrative Example
- Conclusion

Platform-Based Design



Problems for Analysis

- Interaction of different event streams on different resources



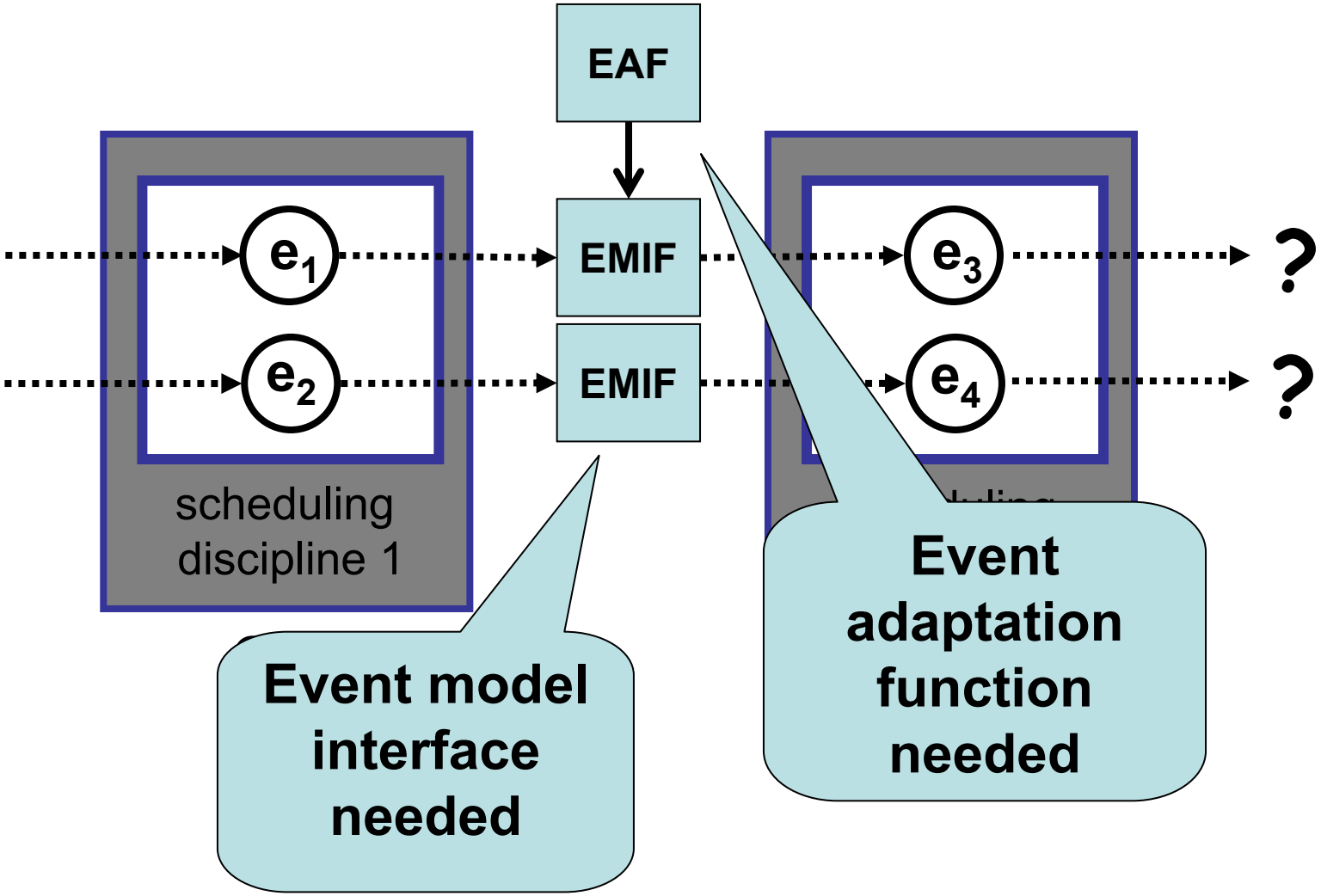
Goals

- Analysis should be composable
- Determine end-to-end properties
- Independent of communication or computation

System-Level Analysis

- Difficult, mostly based on simulation
- Formal analysis recently proposed by [Richter et al., DAC & DATE ‘02]
  - Based on standard event models like periodic, sporadic, with and without jitter…
- A general framework for analysing platform-based design
  - Based on Real-Time Calculus [Thiele et al., ‘00]

EMIFs & EAFs


EAF
EMIF
EMIF
e_1
e_2
e_3
e_4
?
?
scheduling discipline 1
Event model interface needed
Event adaptation function needed

Drawbacks

- In real life: Event streams are not standard (like purely periodic, sporadic, etc.)
  - Errors introduced by approximating event traces
- For each (event stream, scheduling discipline) tuple a different analysis method is needed
- Hardware must be changed in order to enable analysis

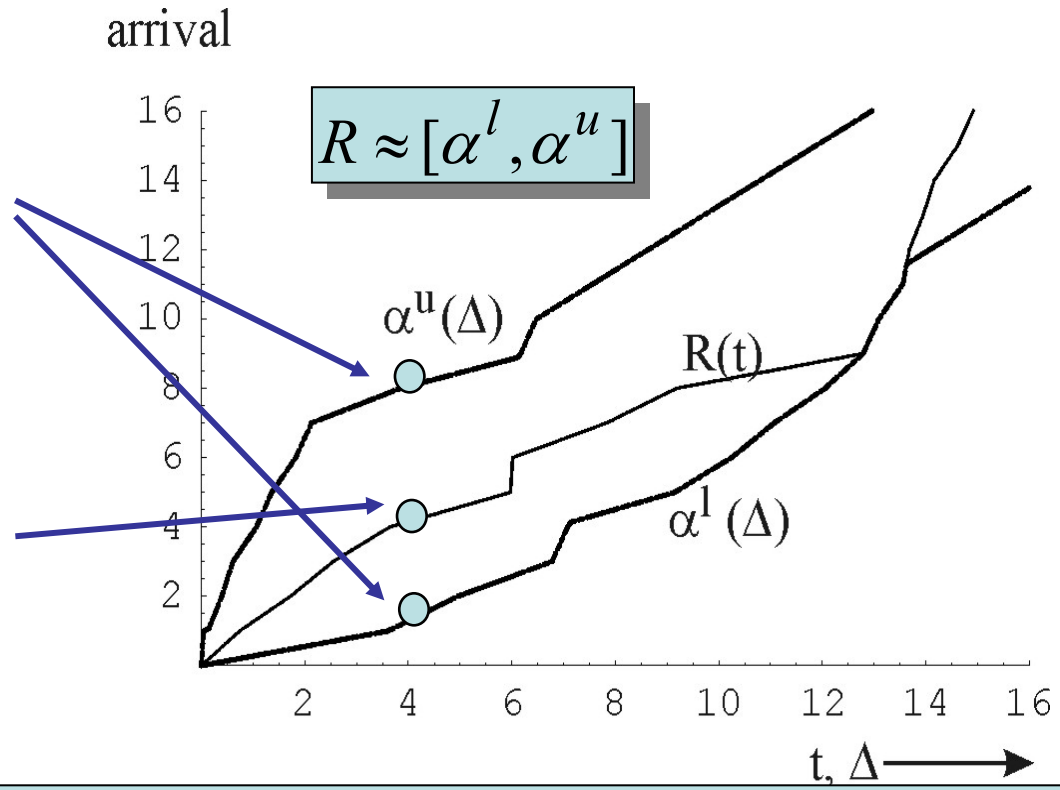
A General Framework

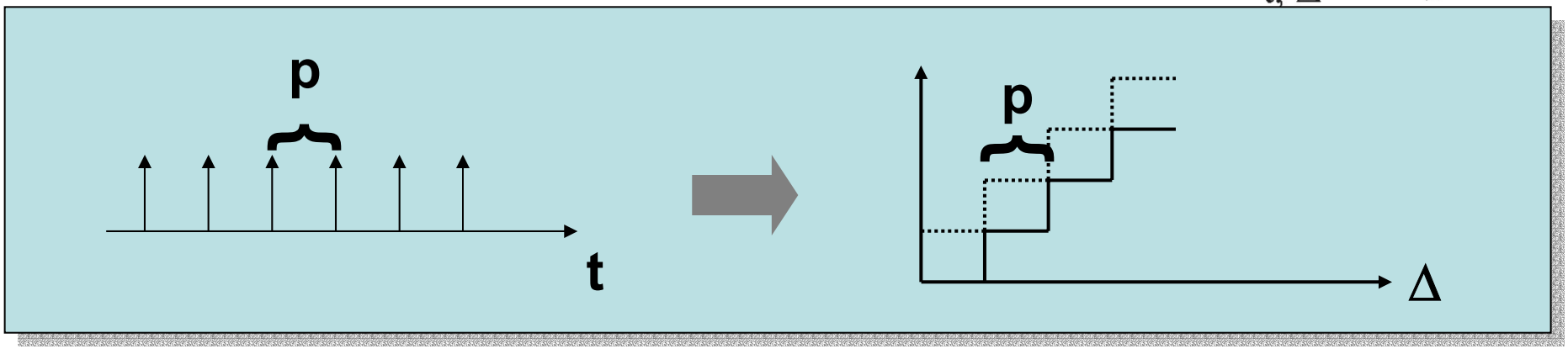
- Event model
- Resource model
- Abstract view of the architecture
- Analysis for single architectural components
- Analysis for interconnections and scheduling policies

Event & Resource Models

maximum/minimum number of events in any interval of length 4

number of events in time interval [0,4]

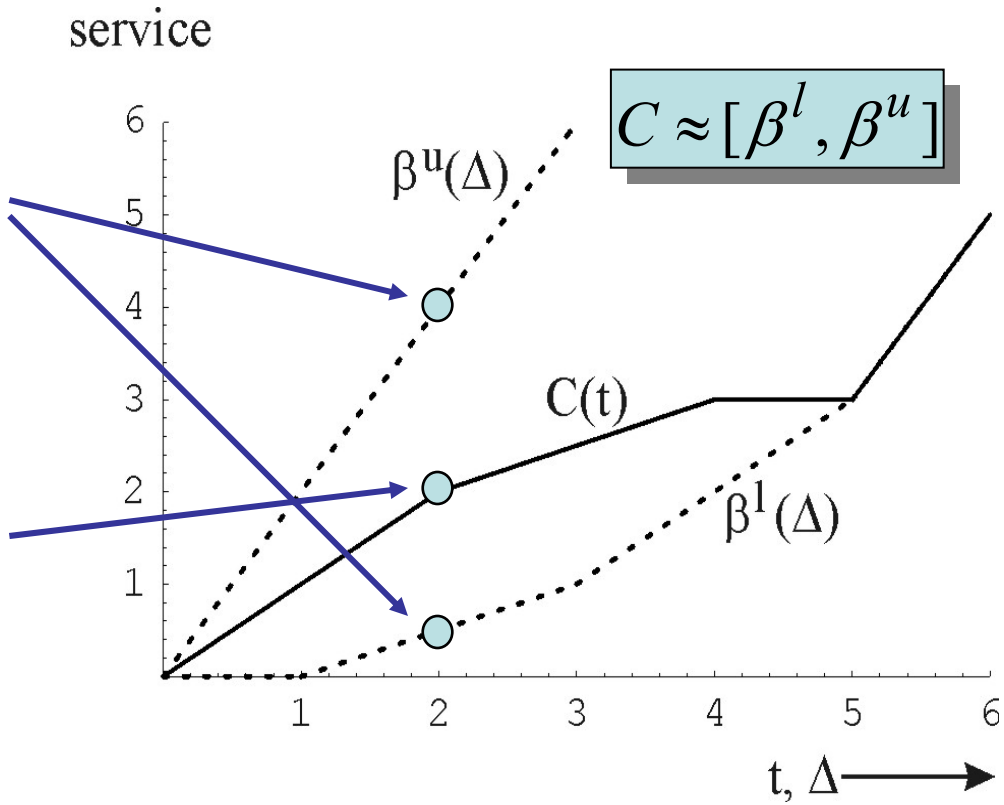




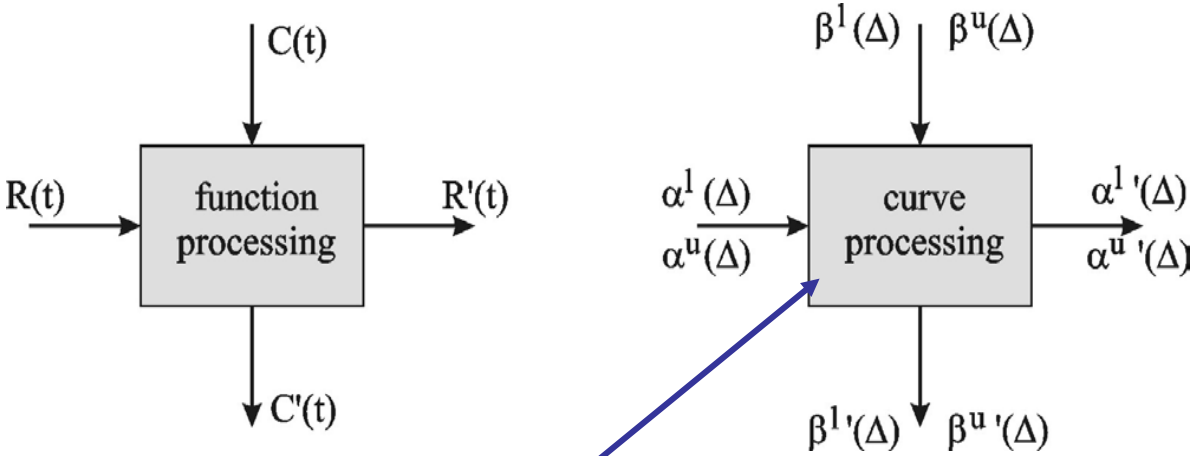
Event & Resource Models (II)

maximum/minimum computing power in any interval of length 2

computing power in time interval [0,2]

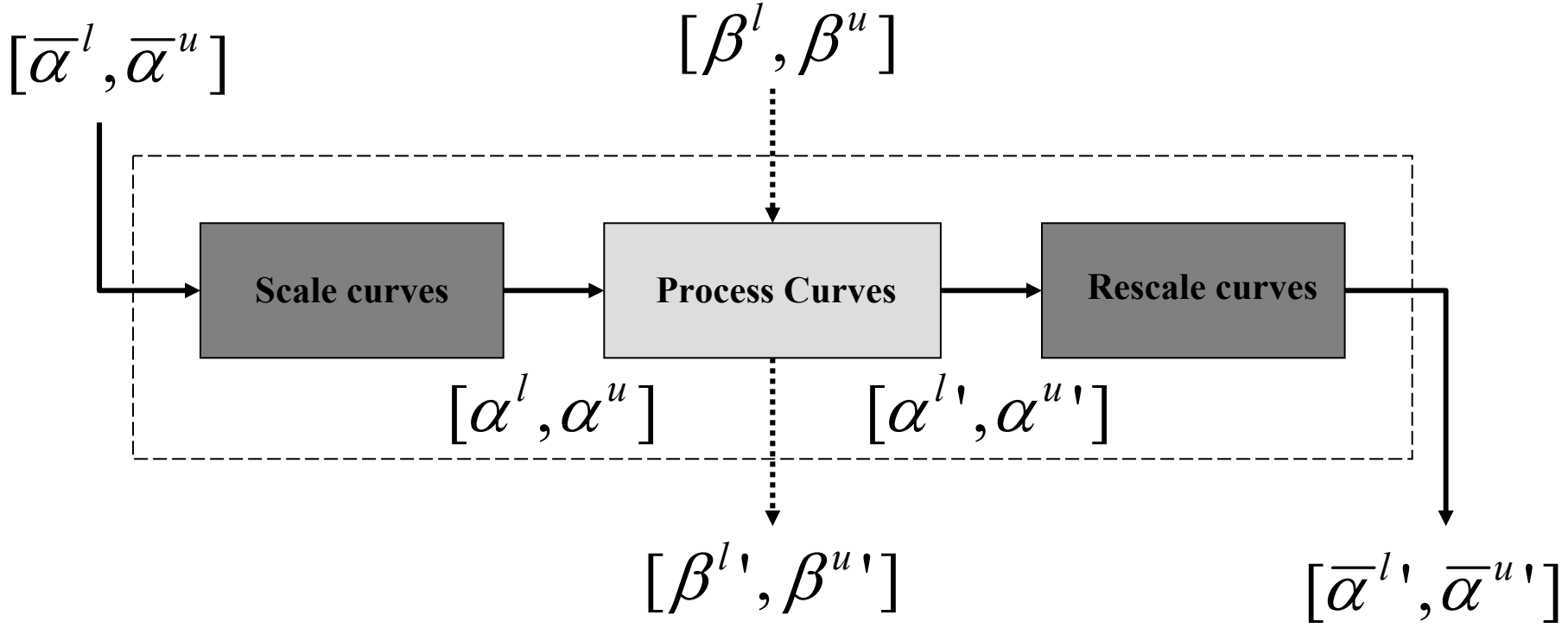


Analysis for a Single Component

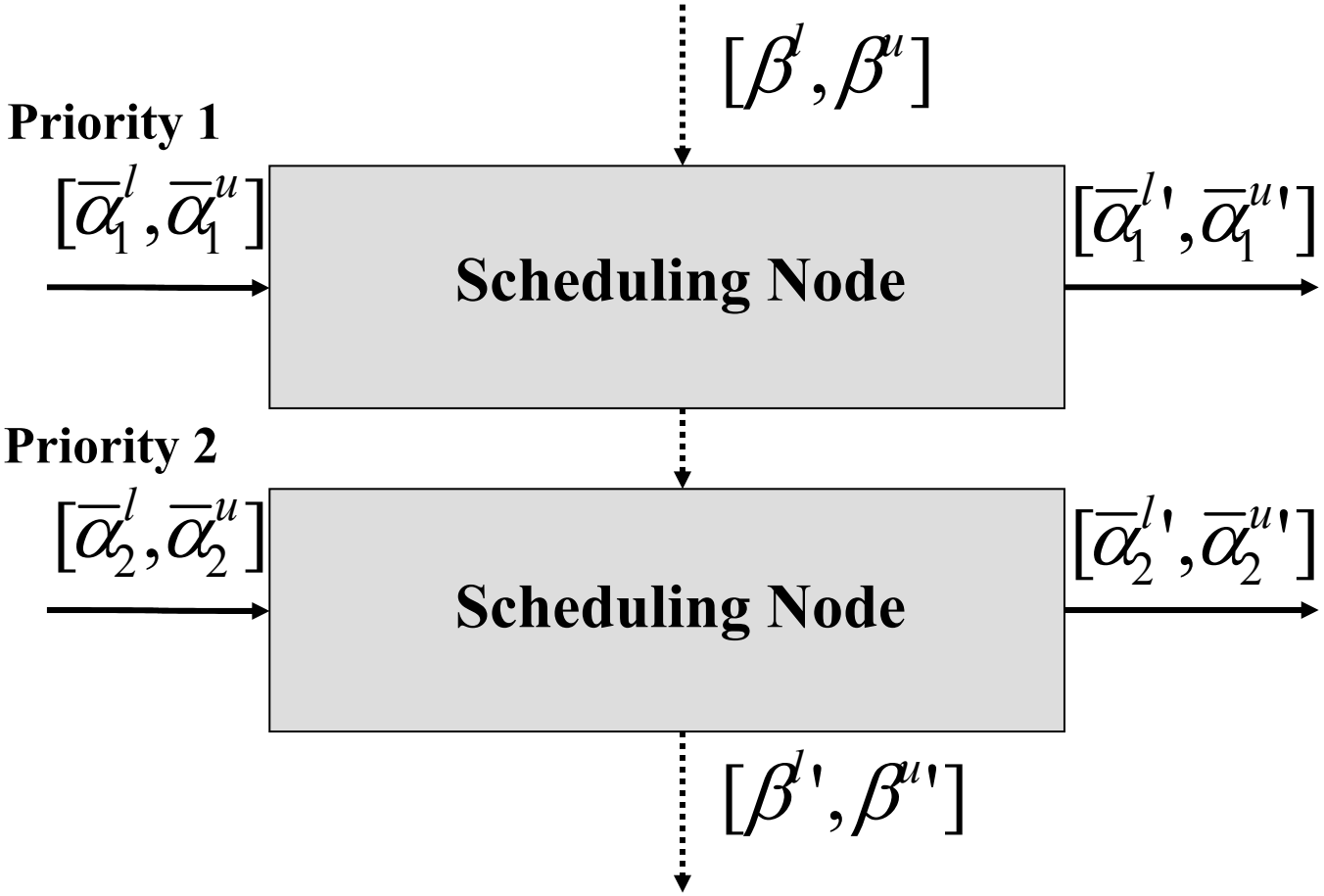


$$
\begin{aligned}
\alpha^{l'}(\Delta) &= \min\{\inf_{0\le\mu\le\Delta}\{\sup_{\lambda\ge 0}\{\alpha^l(\mu+\lambda)-\beta^u(\lambda)\}+\beta^l(\Delta-\mu)\},\beta^l(\Delta)\}\\
\alpha^{u'}(\Delta) &= \min\{\sup_{\lambda\ge 0}\{\inf_{0\le\mu<\lambda+\Delta}\{\alpha^u(\mu)+\beta^u(\lambda+\Delta-\mu)\}-\beta^l(\lambda)\},\beta^u(\Delta)\}\\
\beta^{l'}(\Delta) &= \sup_{0\le\lambda\le\Delta}\{\beta^l(\lambda)-\alpha^u(\lambda)\}\\
\beta^{u'}(\Delta) &= \max\{\inf_{\lambda\ge\Delta}\{\beta^u(\lambda)-\alpha^l(\lambda)\},0\}
\end{aligned}
$$

A Scheduling Node

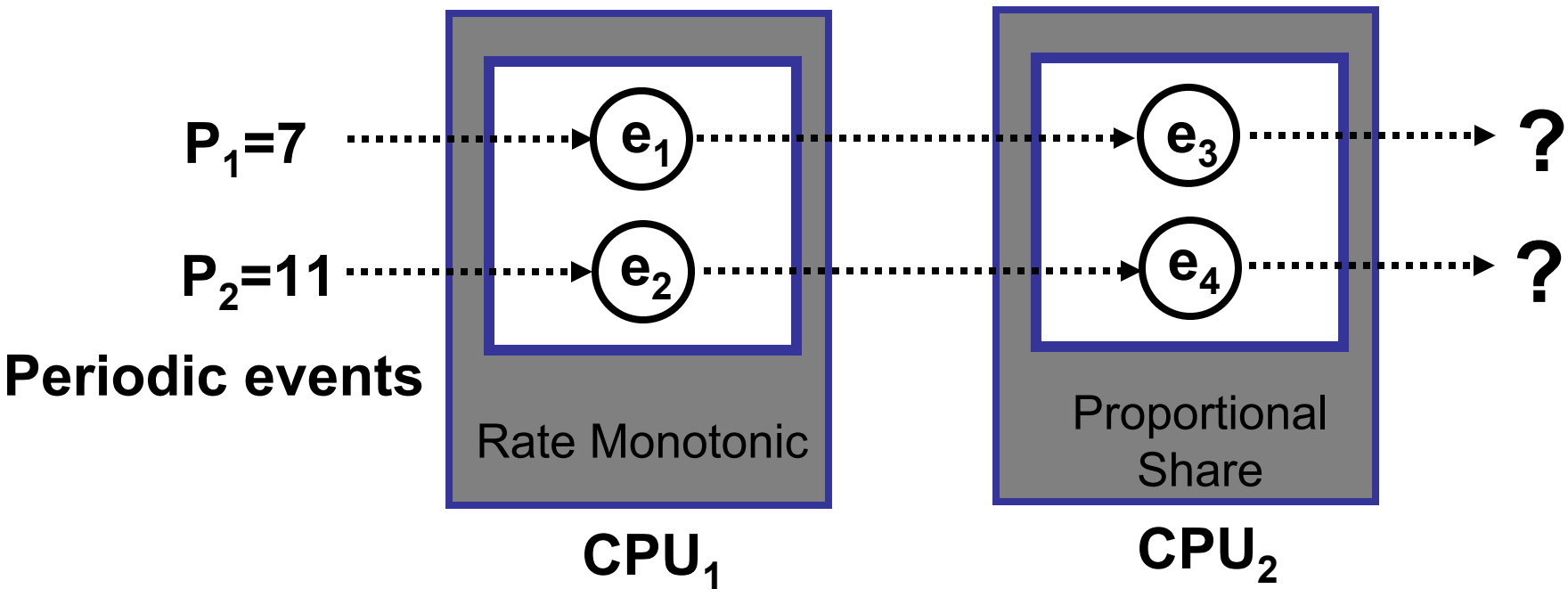


Composing Architectural components



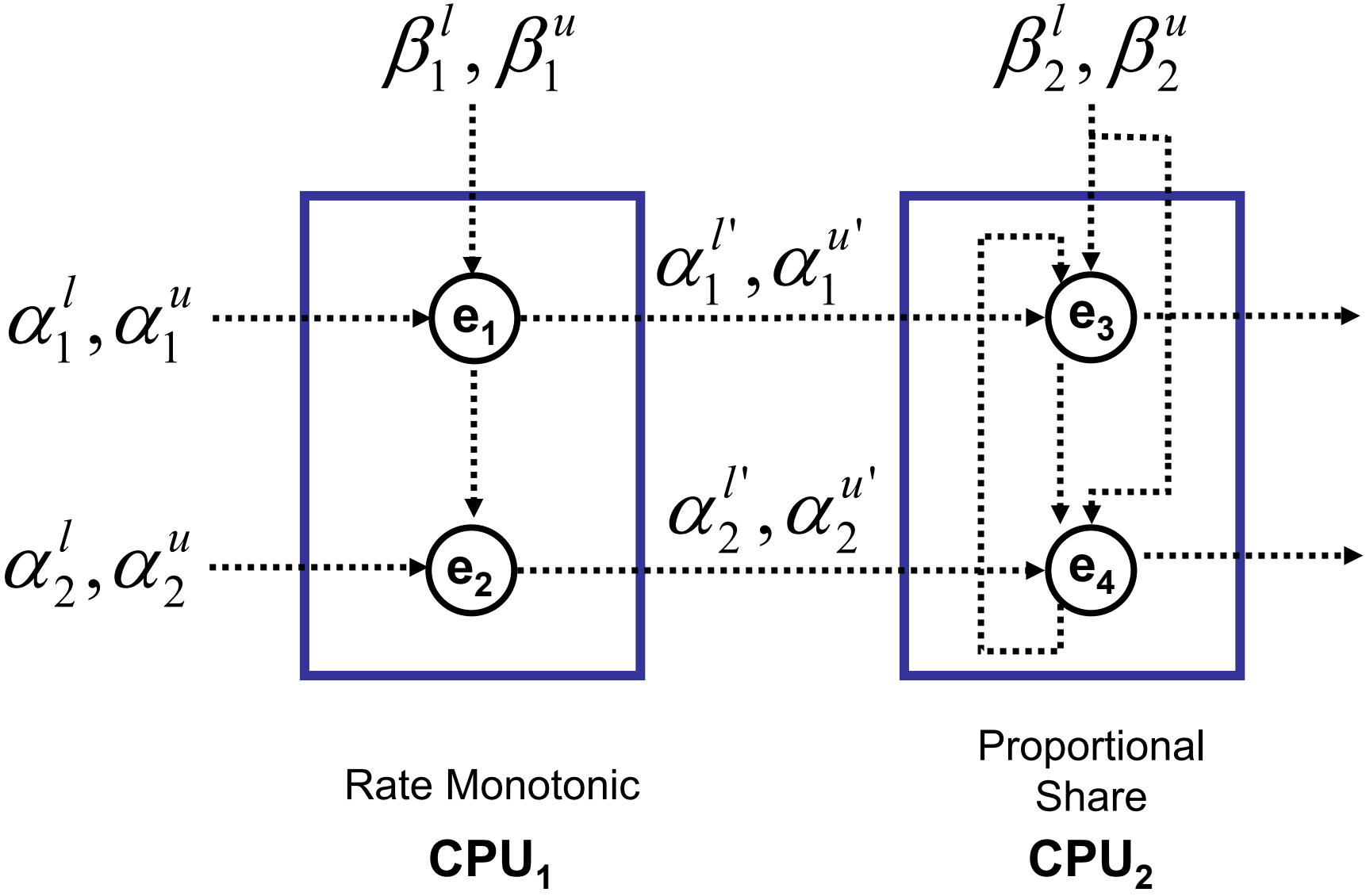
Example scheduling disciplines: Fixed Priority

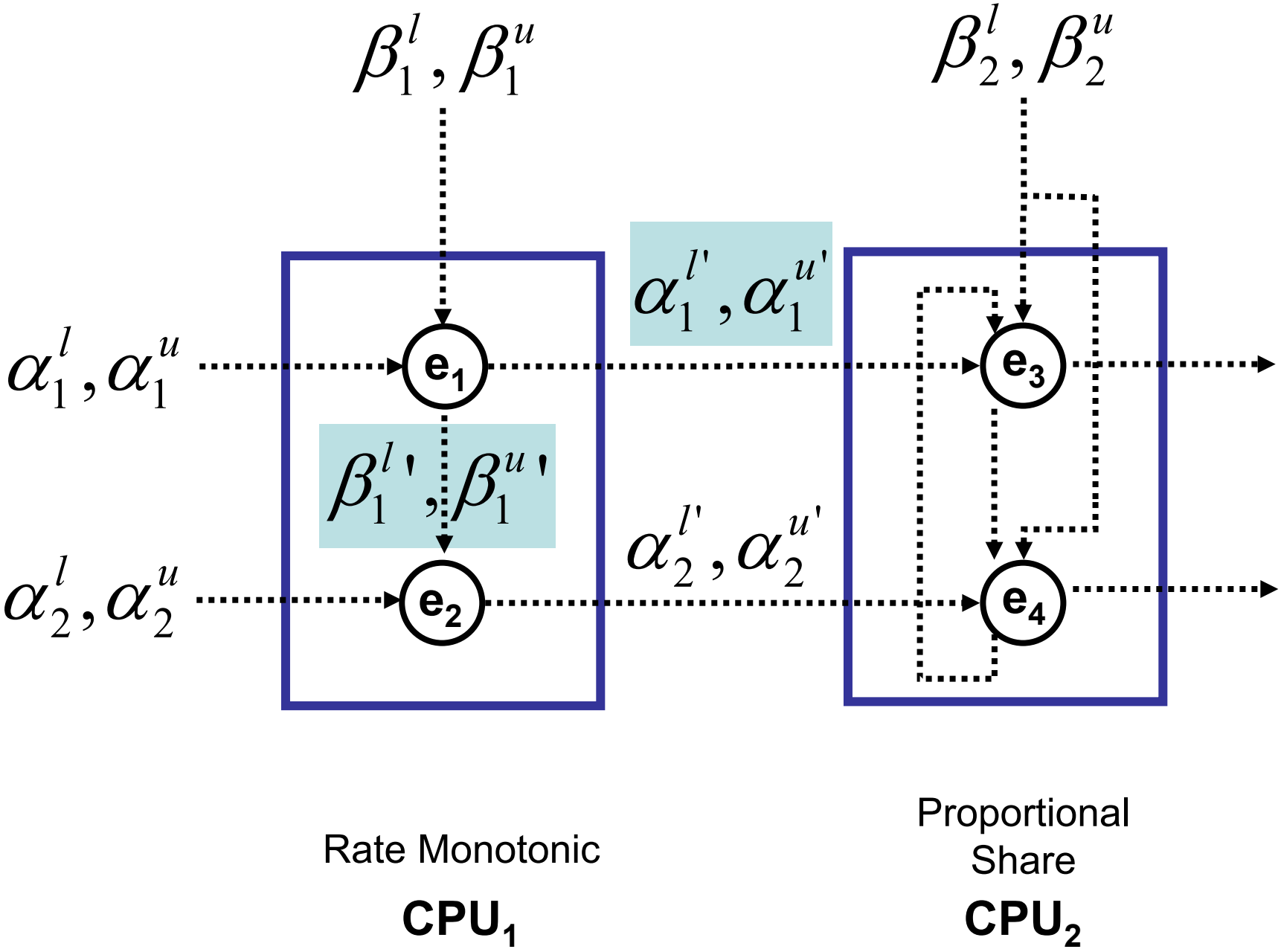
Example: Timing Analysis



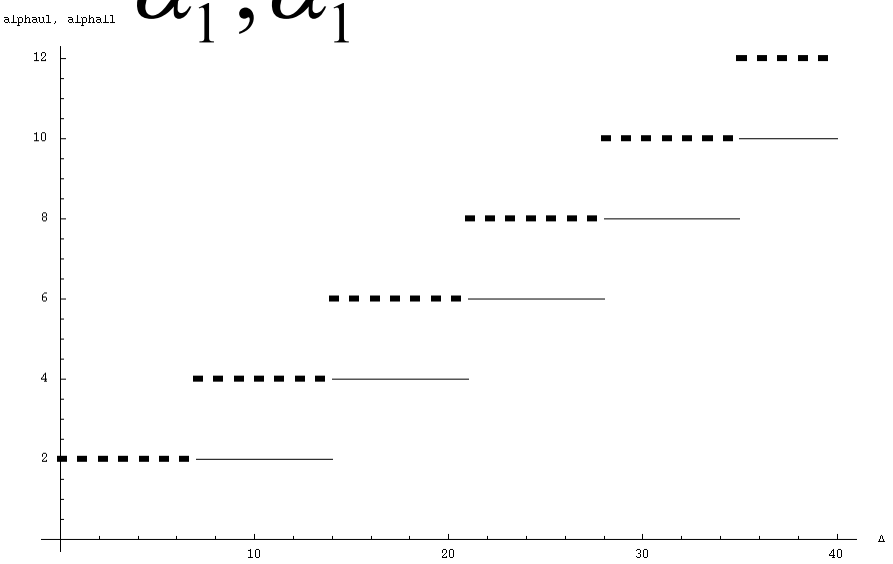
Execution requirement $e_i = 2, \quad i = 1,...,4$

Scheduling Network

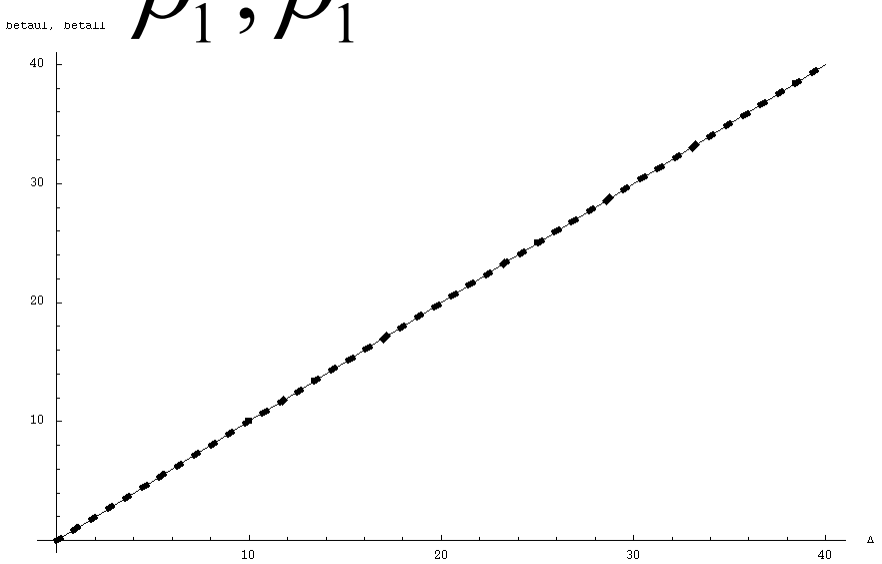



β_1^l, β_1^u
β_2^l, β_2^u
$\alpha_1^{l'}, \alpha_1^{u'}$
α_1^l, α_1^u
e1
e3
$\beta_1^{l'}, \beta_1^{u'}$
$\alpha_2^{l'}, \alpha_2^{u'}$
α_2^l, α_2^u
e2
e4
Rate Monotonic
CPU1
Proportional Share
CPU2

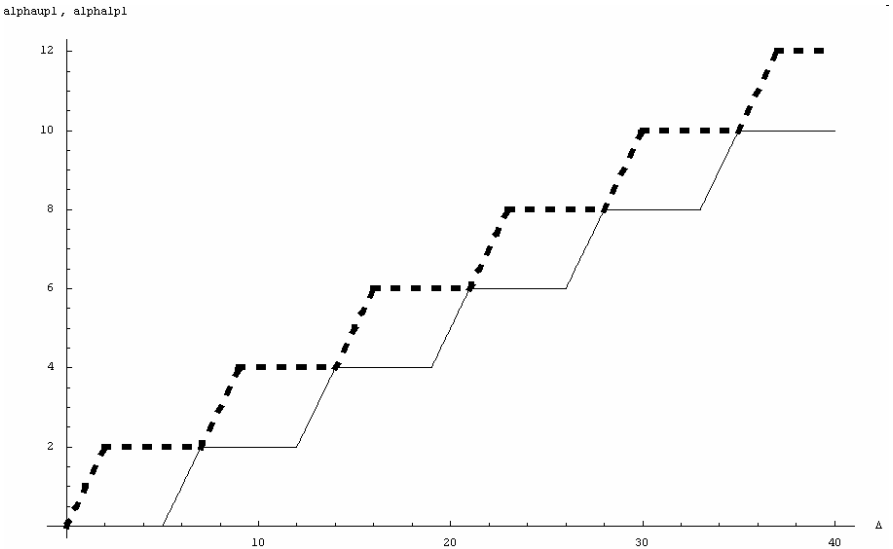
α_1^l, α_1^u



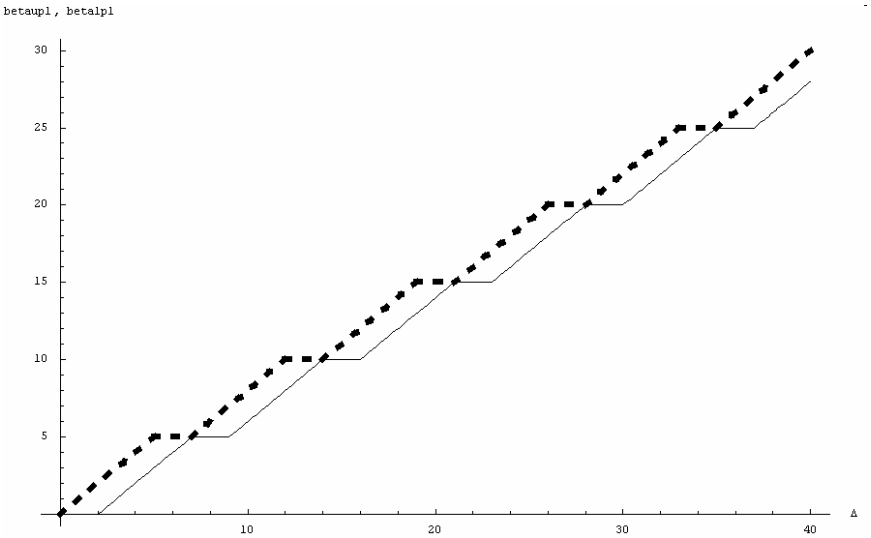
β_1^l, β_1^u

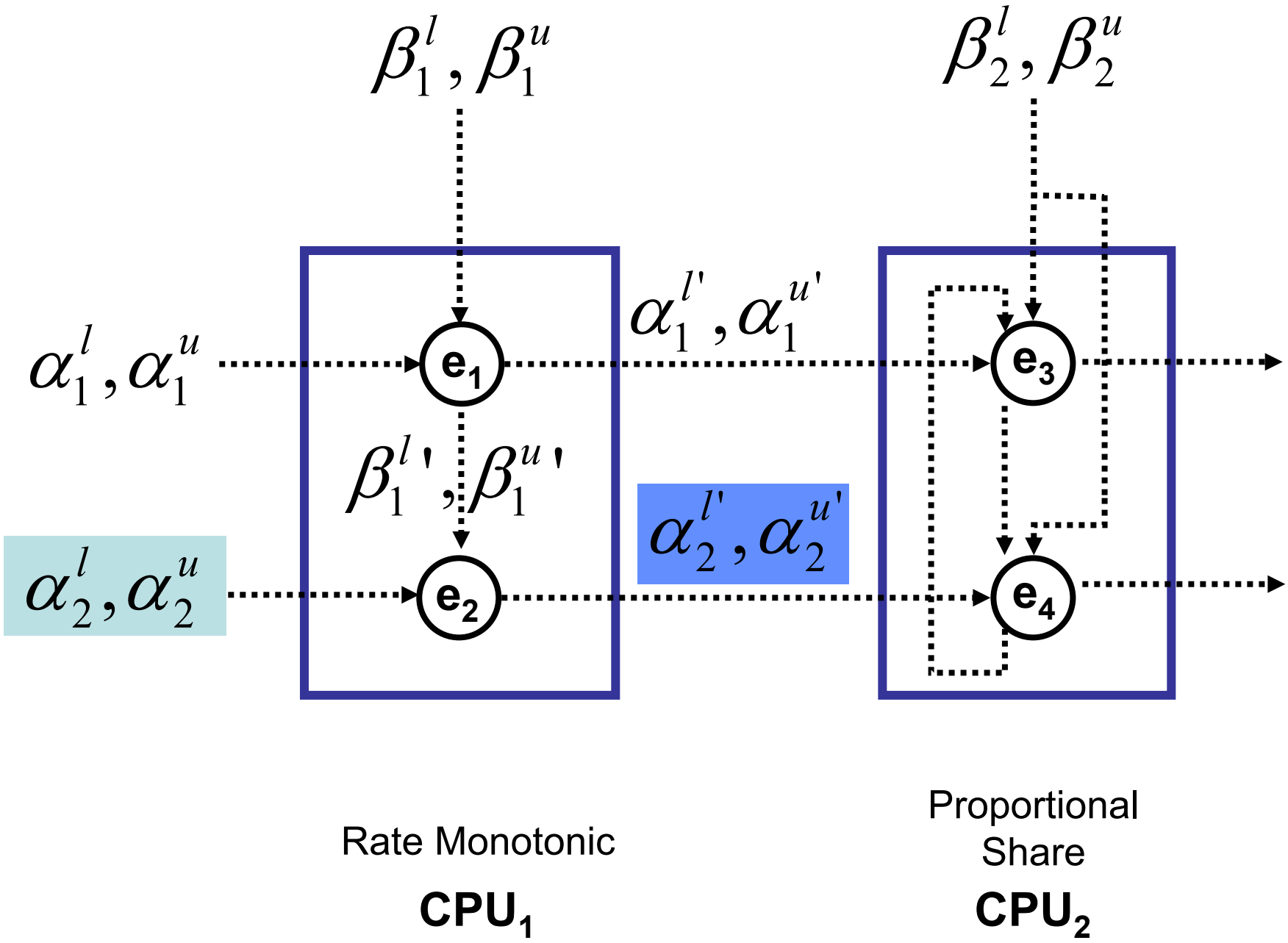


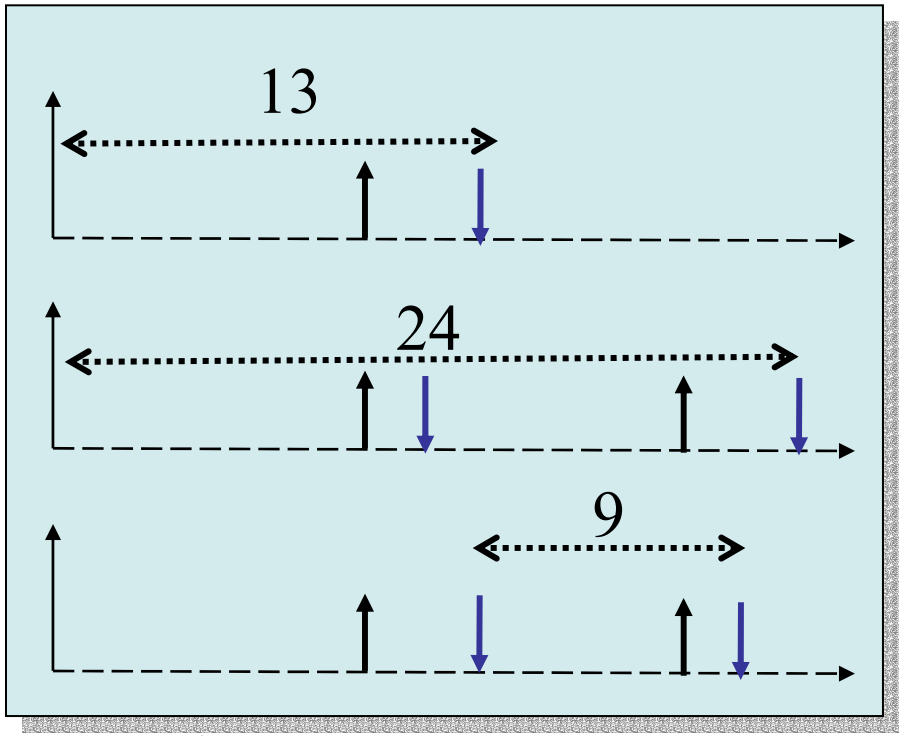
$\alpha_1^{l'}, \alpha_1^{u'}$

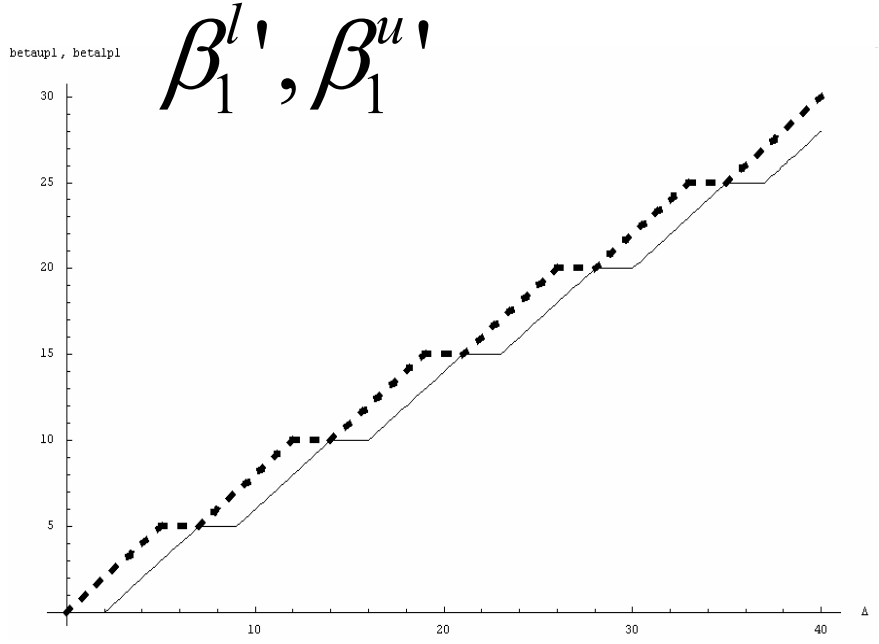


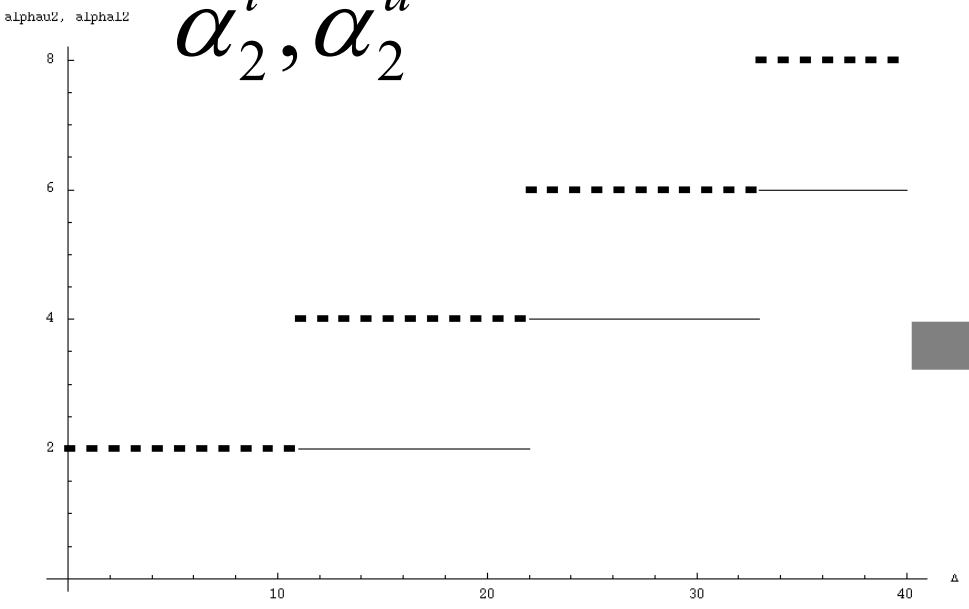
$\beta_1^{l'}, \beta_1^{u'}$

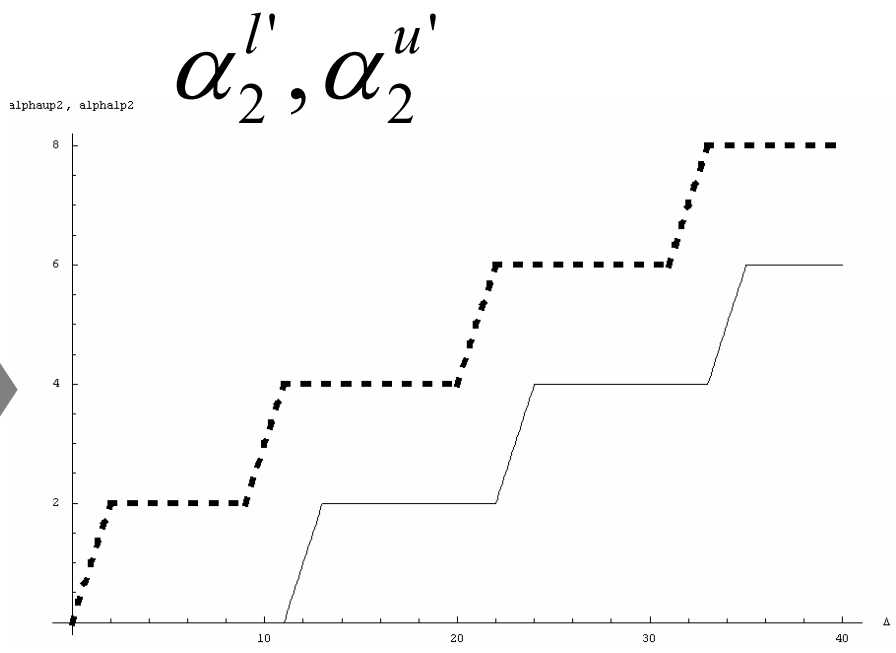


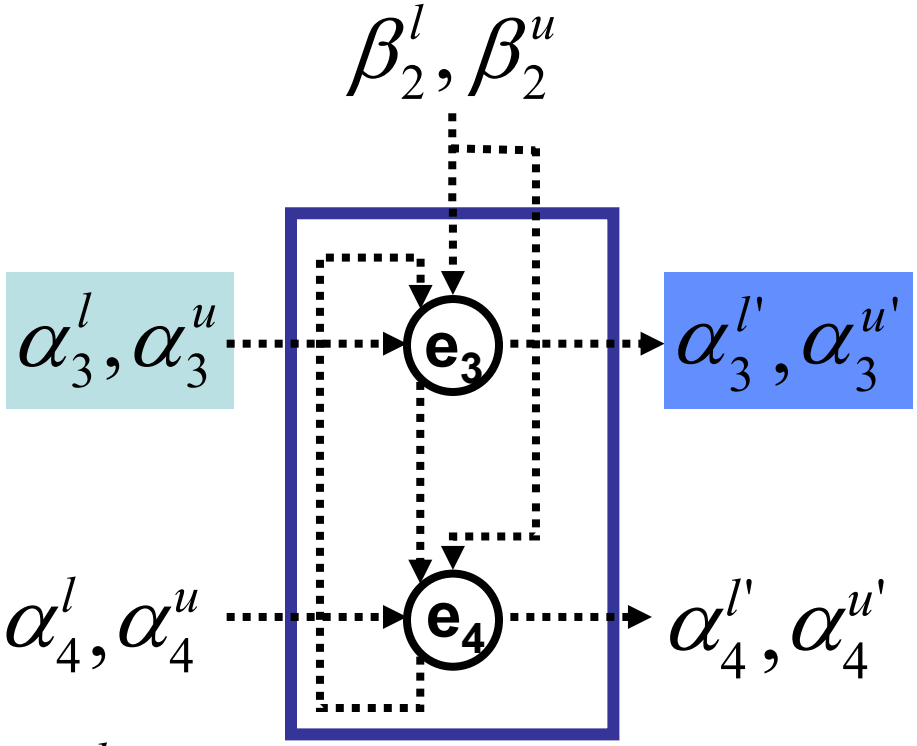

β_1^l, β_1^u
β_2^l, β_2^u
α_1^l, α_1^u
$\alpha_1^{l'}, \alpha_1^{u'}$
$\beta_1^{l'}, \beta_1^{u'}$
$\alpha_2^{l'}, \alpha_2^{u'}$
α_2^l, α_2^u
e1
e2
e3
e4
Rate Monotonic
CPU1
Proportional Share
CPU2

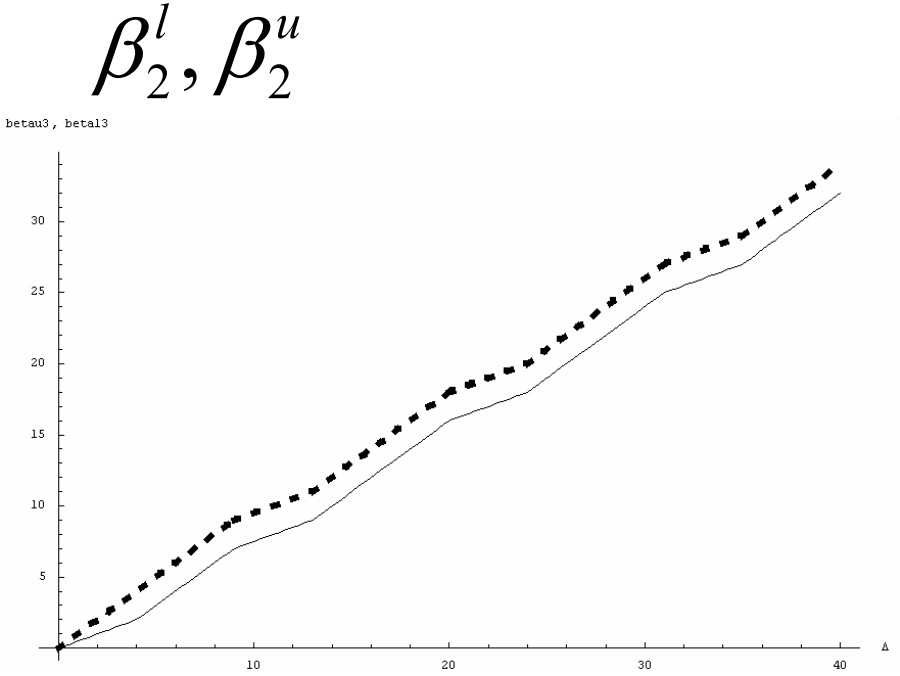

13
24
9

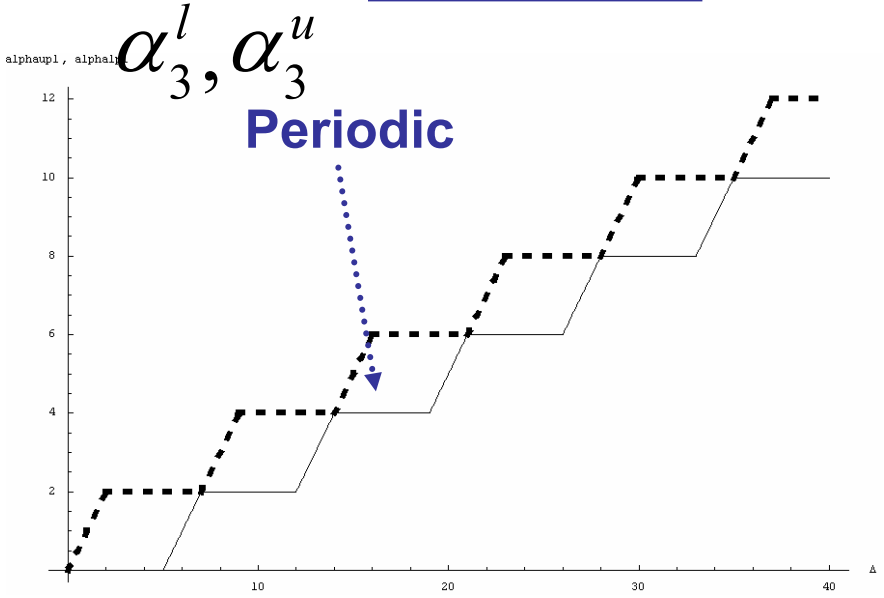

betaup1, betalp1
$\beta_1^{l\,'}, \beta_1^{u\,'}$
30
25
20
15
10
5
10
20
30
40
Δ

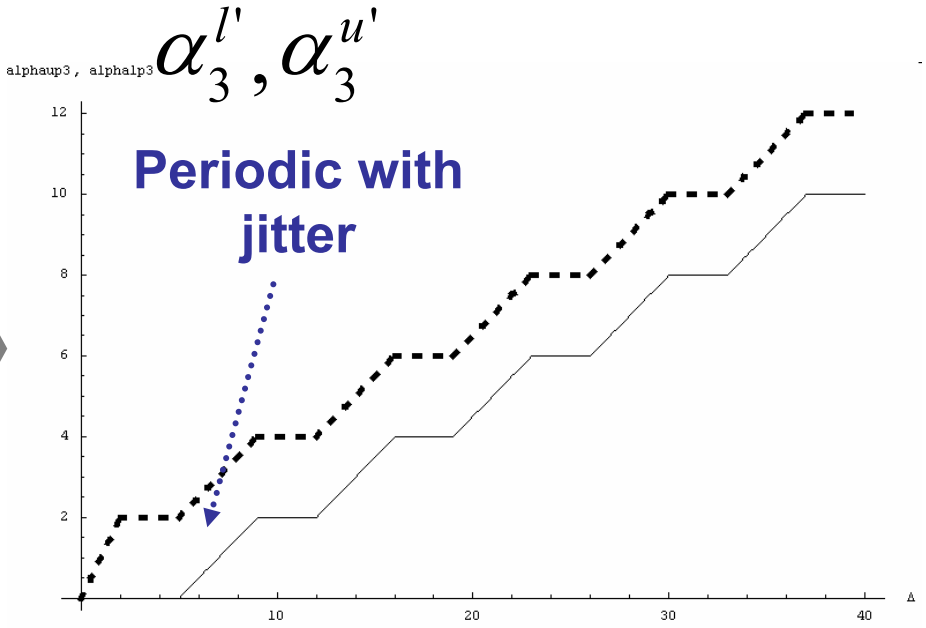

alphau2, alphal2
α_2^l, α_2^u
8
6
4
2
10
20
30
40
Δ


alphaup2, alphalp2
$\alpha_2^{l'}, \alpha_2^{u'}$
8
6
4
2
10
20
30
40
Δ


β_2^l, β_2^u
α_3^l, α_3^u
e3
$\alpha_3^{l'}, \alpha_3^{u'}$
α_4^l, α_4^u
e4
$\alpha_4^{l'}, \alpha_4^{u'}$


β_2^l, β_2^u
betau3, betal3
30
25
20
15
10
5
10
20
30
40
Δ


α_3^l, α_3^u
alphaup1, alphalp
Periodic
12
10
8
6
4
2
10
20
30
40
Δ


$\alpha_3^{l'}, \alpha_3^{u'}$
alphaup3, alphalp3
Periodic with jitter
12
10
8
6
4
2
10
20
30
40
Δ

Conclusion

- Generalised system analysis method for platform-based systems
- Not restricted to standard event models
- Extendable by new scheduling policies
- Provably tight bounds