ETH Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Computer Engineering and Networks Laboratory TIK

Combining Optimistic and Pessimistic DVS Scheduling: An Adaptive Scheme and Analysis

Simon Perathoner [1] Jian-Jia Chen [2] Kai Lampka [1] Nikolay Stoimenov [1] Lothar Thiele [1]

[1] ETH Zurich (Switzerland) [2] KIT Karlsruhe (Germany)

ICCAD, November 8th, 2010, San Jose, CA

Outline

- **Motivation**
- Background
- Example
- Proposed Adaptive DVS Scheme
- Design and Verification of Adaptive Scheme
- Experimental Evaluation
- Conclusions

Embedded Systems

Computer systems physically embedded into larger device

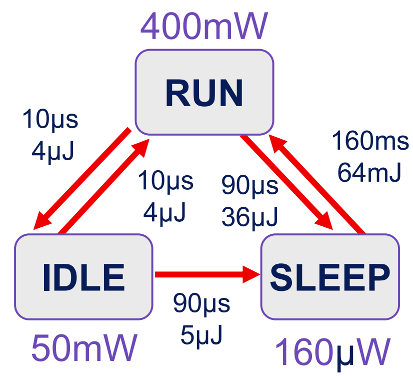
Requirements

- **Performance**
- **Energy**
- **Fault tolerance**
- **Size**
- **Weight**
- **Cost**

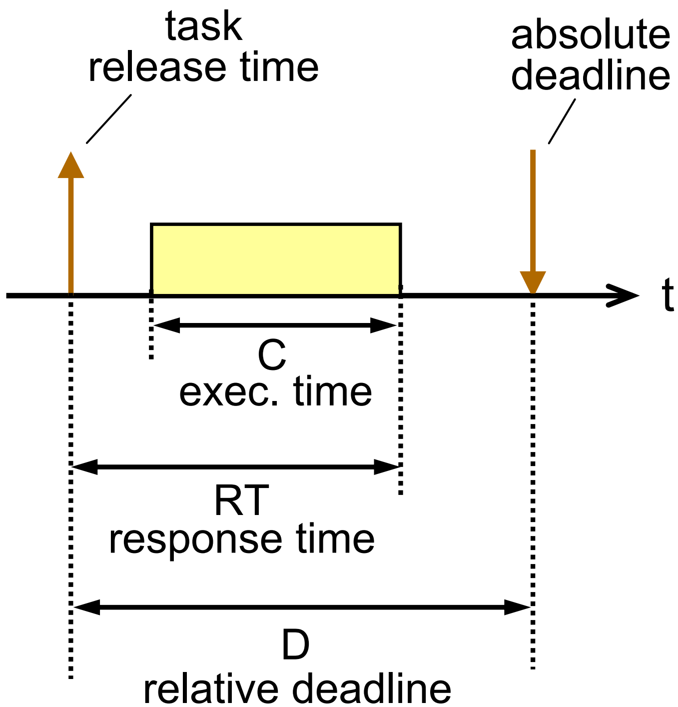
Methods for CPU Power Management

- **Dynamic Voltage Scaling (DVS)**
  - For CMOS circuits $E \propto V_{dd}^2$ (ignoring leakage)
  - Save energy by reducing supply voltage (clock frequency)

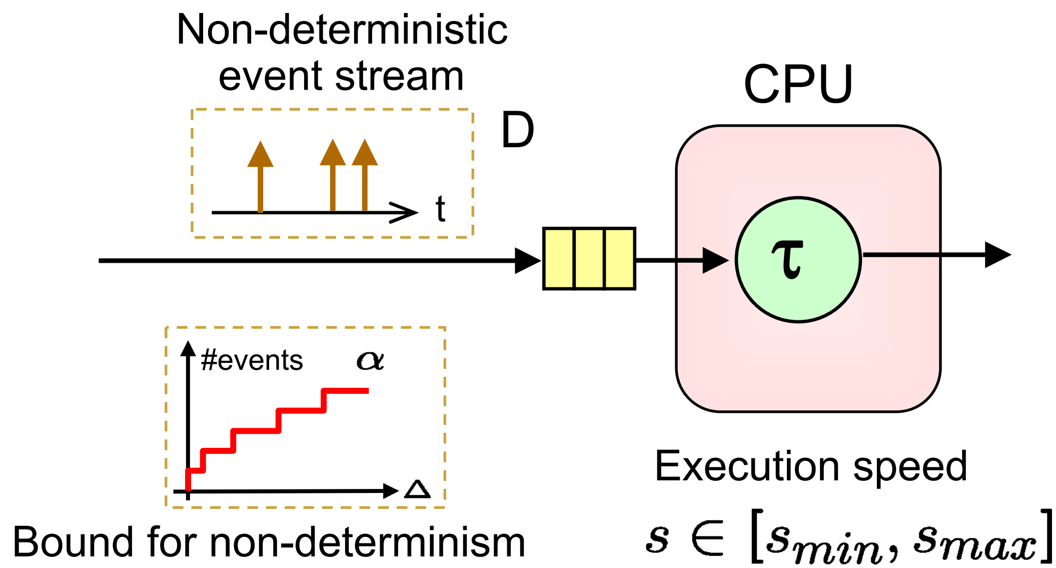
- **Dynamic Power Management (DPM)**
  - Switch between different power states
  - Helps to reduce power dissipation due to leakage



System Model



$$WCRT \leq D$$



Goal: Minimize energy consumption while guaranteeing deadlines

Related Work

- **Offline DVS Approaches**
  - Take scheduling decisions statically (at design time) based on expected worst-case workload
  - E.g. [Yao et al. 1995], [Quan et al. 2007], [Maxiaguine et al. 2005]
  - Problem: If the actual event trace differs from the worst-case, the execution speed is unnecessarily high!

  → **They are often too pessimistic (waste energy)**

- **Online DVS Approaches**
  - Take scheduling decisions dynamically (at run time) by adapting to actual workload
  - E.g. [Yao et al. 1995], [Aydin et al. 2001], [Bansal et al. 2005]
  - Problem: They may go above the maximum available speed!

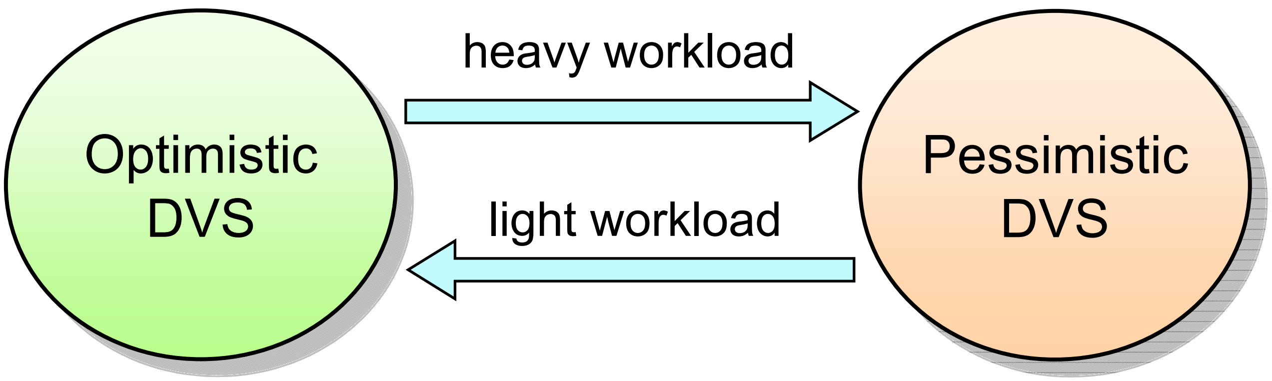
  → **They may be too optimistic (improvident)**

Idea

Adaptive scheme which combines Online and Offline DVS

Idea: Apply optimistic online DVS if system is light-loaded and pessimistic offline DVS if system is heavy-loaded

Key issue: Decide when to switch between the two modes



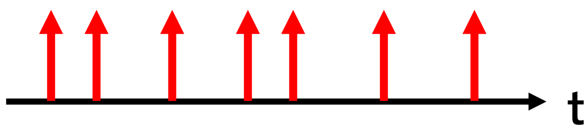
Outline

- Motivation
- **Background**
- Example
- Proposed Adaptive DVS Scheme
- Design and Verification of Adaptive Scheme
- Experimental Evaluation
- Conclusions

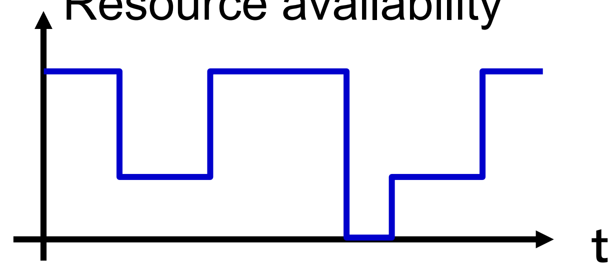
Event and Resource Models

Real-Time Calculus (RTC) [Thiele et al. 2000]

Event stream

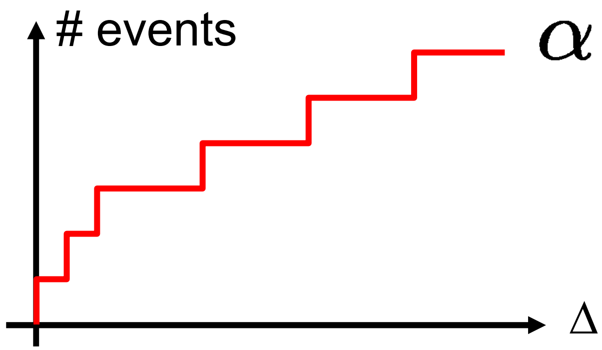


Resource availability



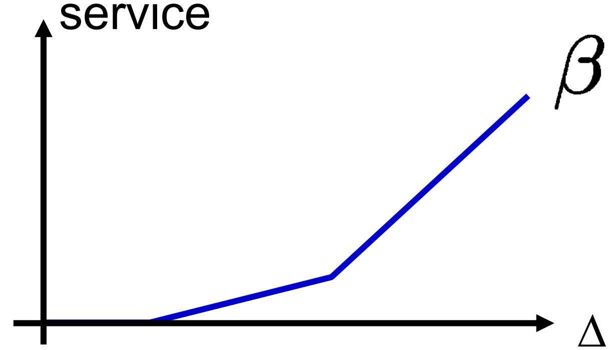
Event model

Service model



Arrival curve

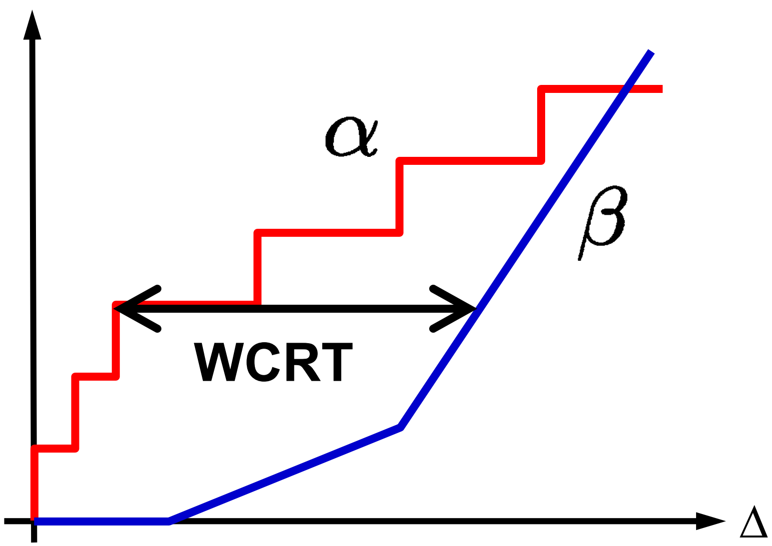
service



Service curve

Pessimistic Offline DVS Scheduling

Scheduling analysis in RTC

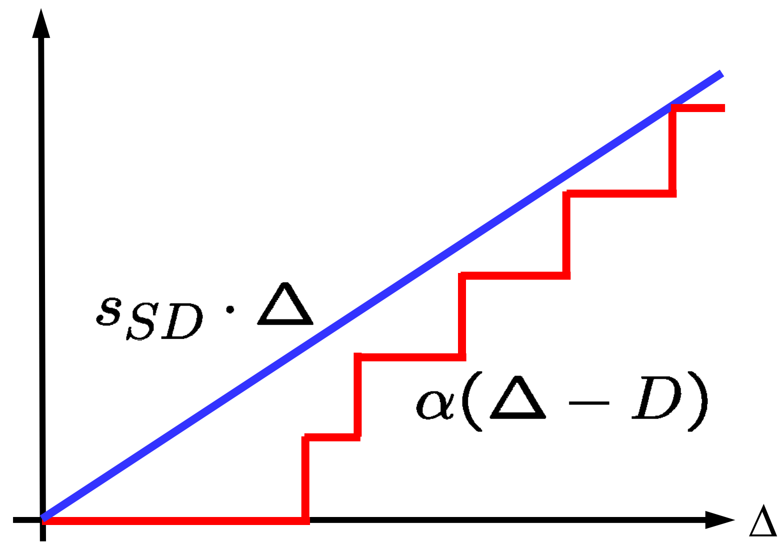


Deadlines guaranteed if

$$WCRT(\alpha, \beta) \leq D$$

$$\alpha(\Delta - D) \leq \beta(\Delta) \quad \forall \Delta \geq 0$$

Pessimistic DVS Scheduling (SD)



Find smallest s_{SD} such that

$$\alpha(\Delta - D) \leq s_{SD} \cdot \Delta \quad \forall \Delta \geq 0$$

Optimistic Online DVS Scheduling

OPT Algorithm [Yao Demers Shenker 1995]

- Greedily select minimum speed that guarantees all deadlines considering only arrived events
- Event-driven algorithm: Speed changes only at event arrival or completion

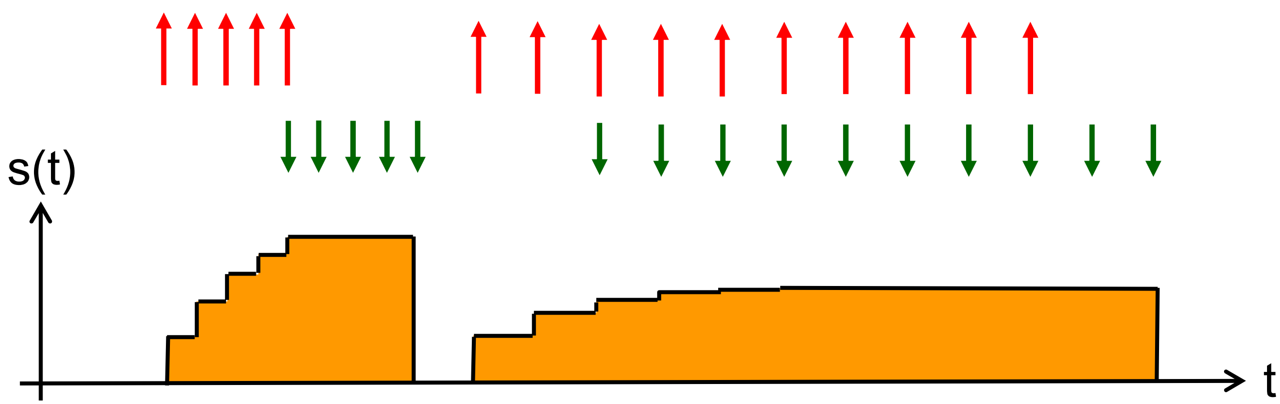
$$s(t) = \max\left\{ s'_{\min}, \max_{e_j}\left\{ \sum_{e_i: a_i \leq t, e_i \preceq e_j} \frac{C_i(t)}{d_j - t} \right\} \right\}$$

C … remaining execution time

a … arrival time

d … deadline (abs.)

$\preceq$ … priority order



Analytical bound for max. speed: [Chen et al. 2009]

Outline

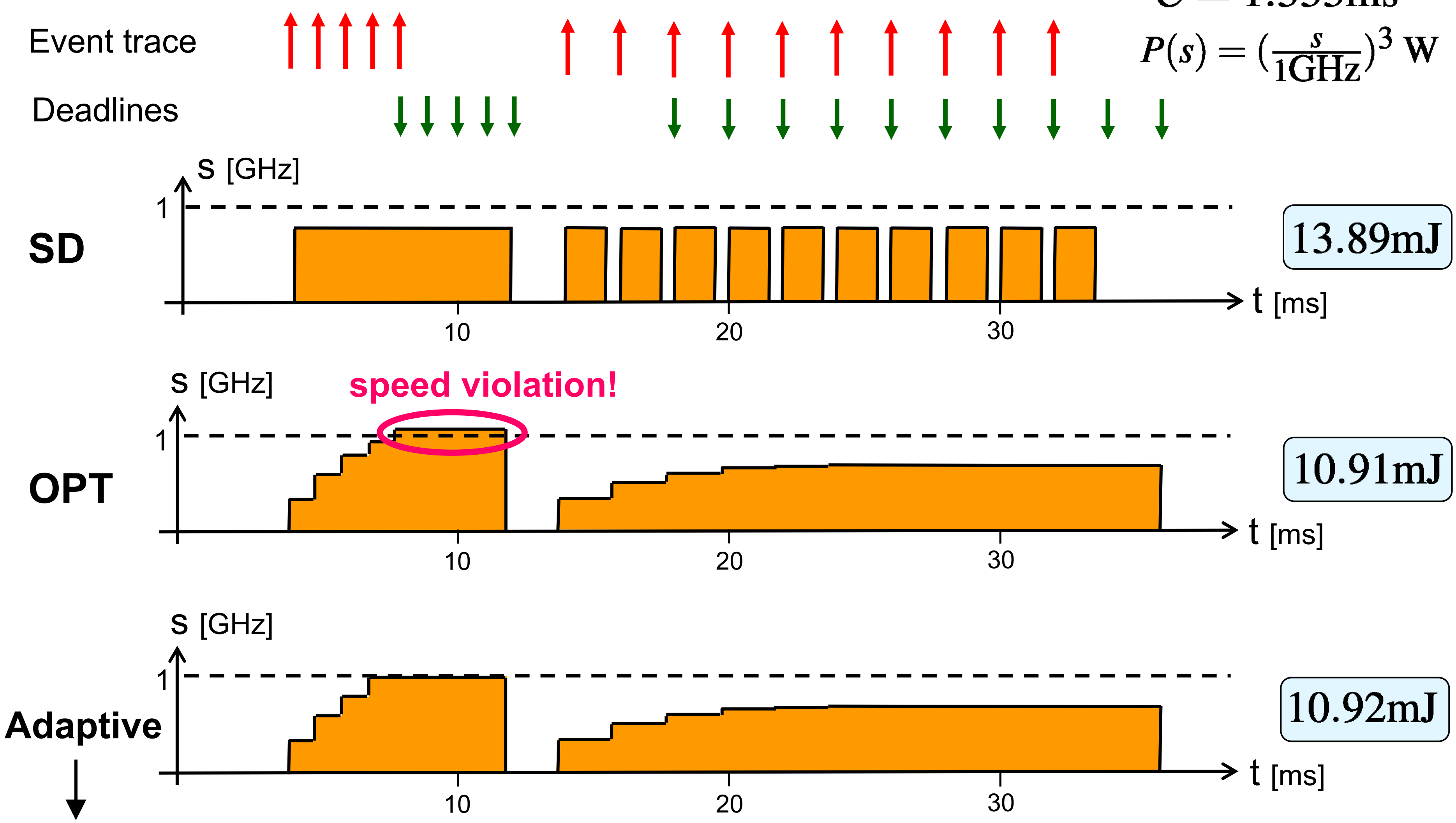
- Motivation
- Background
- **Example**
- Proposed Adaptive DVS Scheme
- Design and Verification of Adaptive Scheme
- Experimental Evaluation
- Conclusions

Motivational Example

$s_{max} = 1\text{GHz}$

$C = 1.333\text{ms}$

$P(s) = (\frac{s}{1\text{GHz}})^3 \text{ W}$



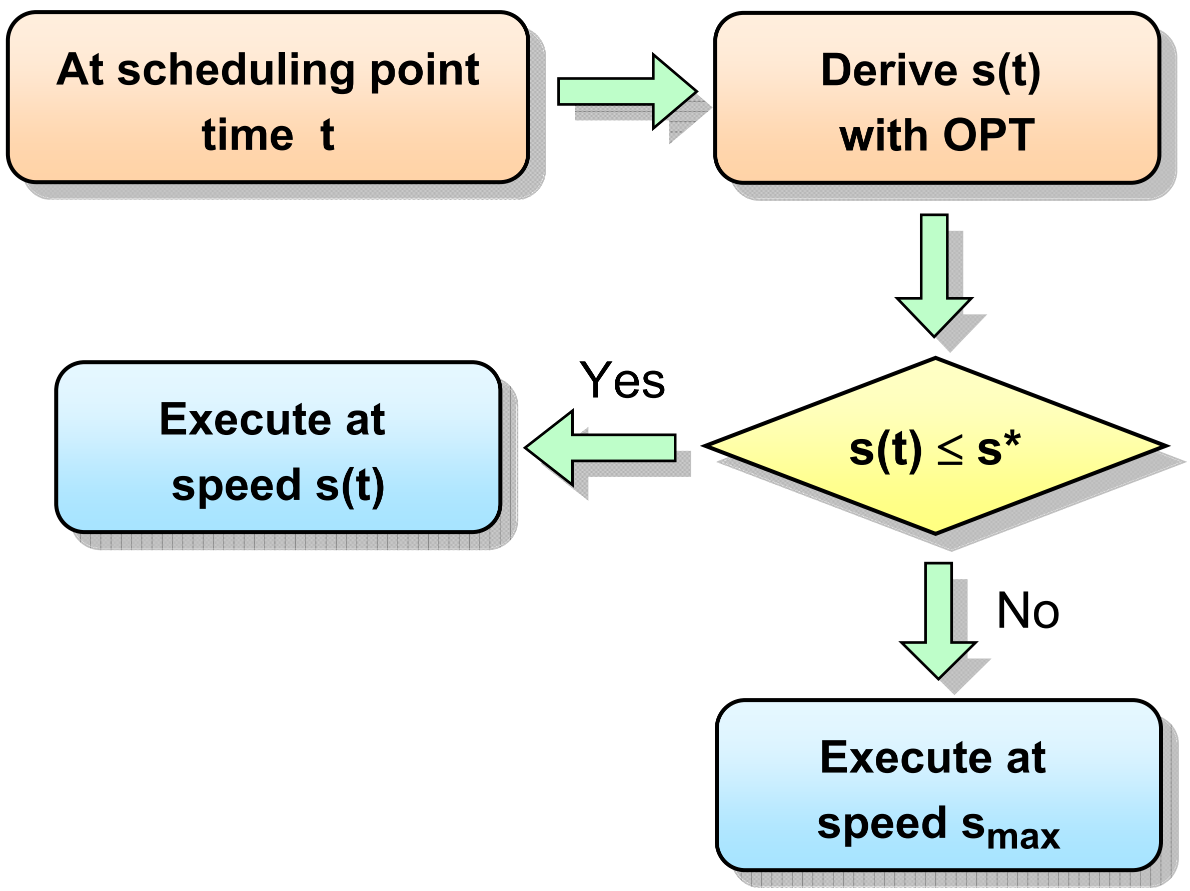
OPT if s < 0.85 GHz, s_{max} if s ≥ 0.85 GHz

Outline

- Motivation
- Background
- Example
- **Proposed Adaptive DVS Scheme**
- Design and Verification of Adaptive Scheme
- Experimental Evaluation
- Conclusions

Adaptive DVS Scheme

s^* = Threshold speed

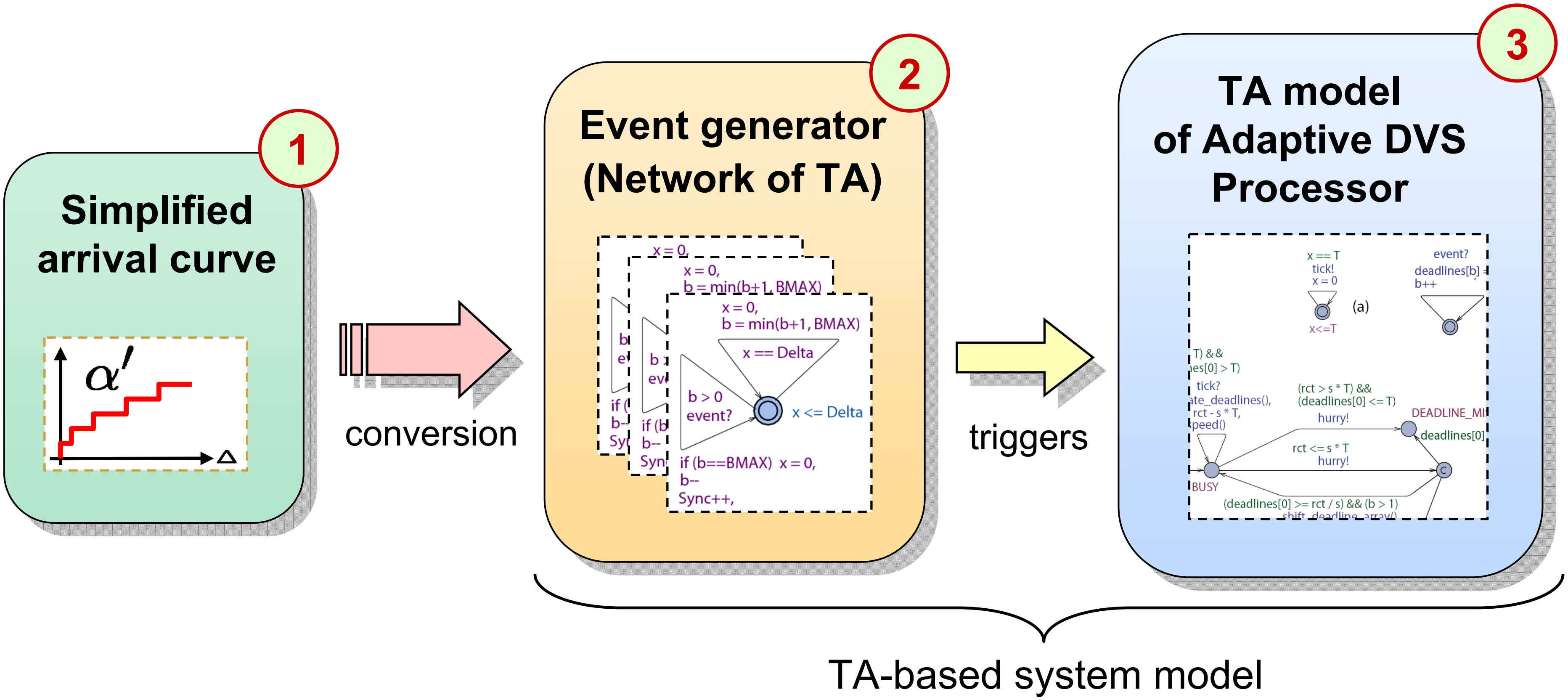


Outline

- Motivation
- Background
- Example
- Proposed Adaptive DVS Scheme
- **Design and Verification of Adaptive Scheme**
- Experimental Evaluation
- Conclusions

Design and Verification - Overview

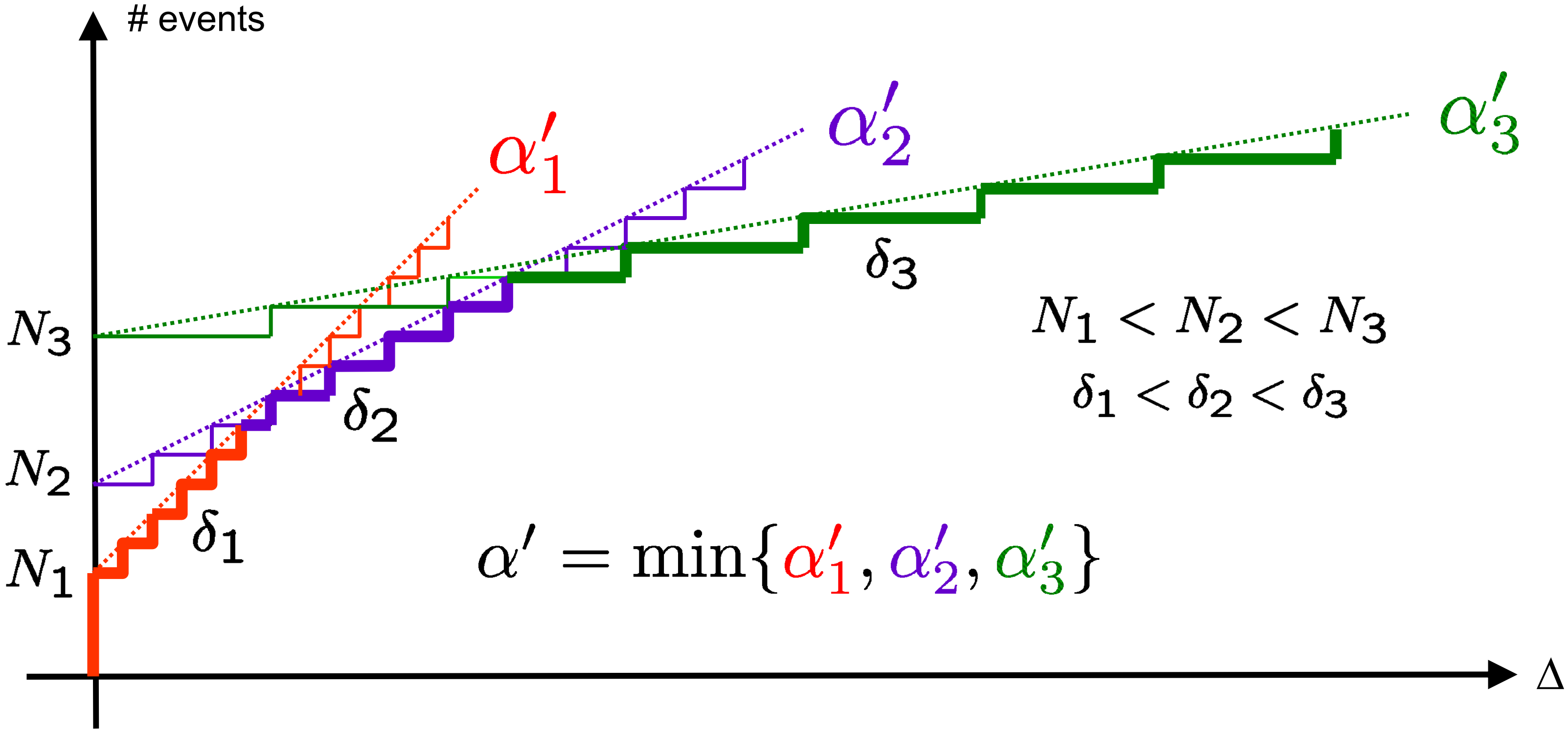
Approach based on hybrid analysis method of [Lampka et al. 2009]



Use model checker UPPAAL and binary search to determine max. s*

1 Simplified Arrival Curves

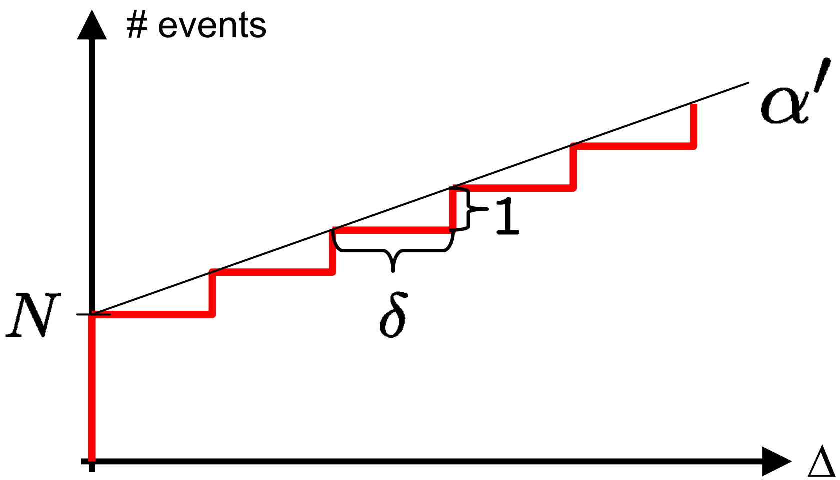
Pseudo-concave arrival curve (increasing step widths)



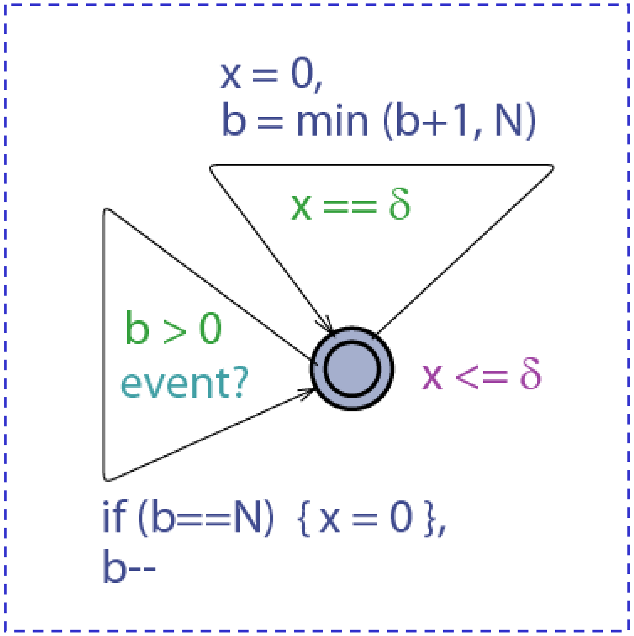
$$\alpha'_i(\Delta) := N_i + \left\lfloor \frac{\Delta}{\delta_i} \right\rfloor \qquad \alpha'(\Delta) := \min_i\{\alpha'_i(\Delta)\}$$

2 TA Representation of Arrival Curve

Arrival curve



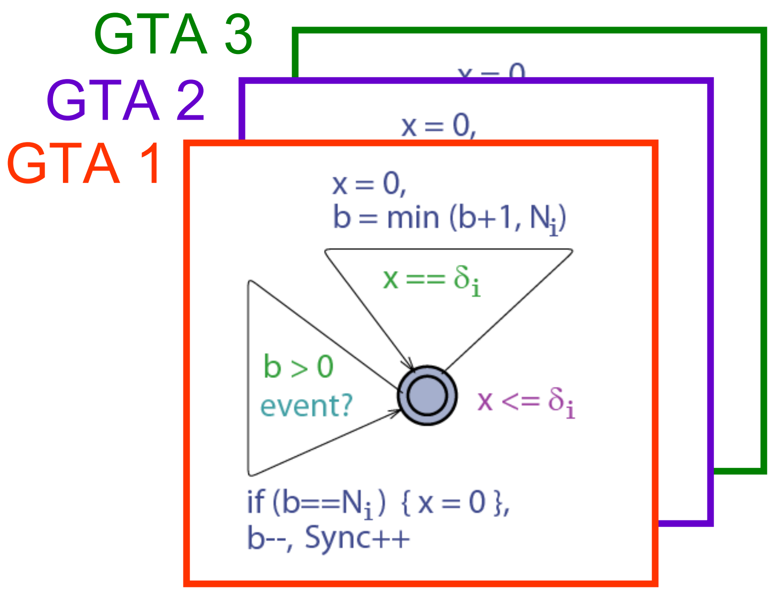
GTA

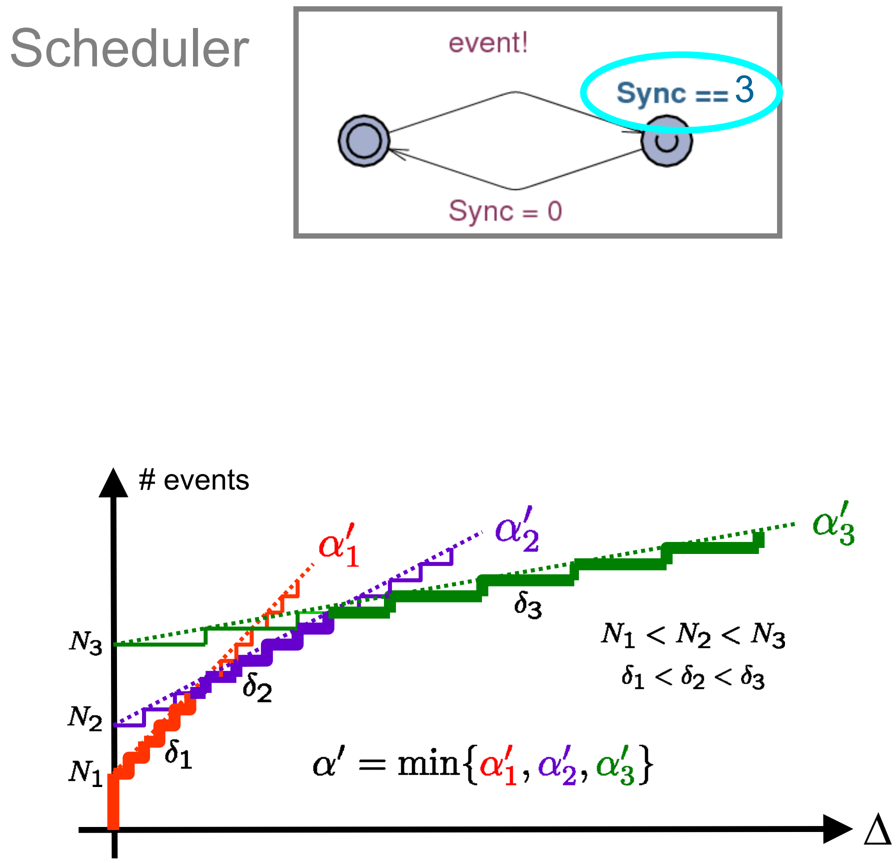


$$\alpha'(\Delta) = N + \left\lfloor \frac{\Delta}{\delta} \right\rfloor$$

GTA guarantees that event stream conforms to α'

2 TA Representation of Arrival Curve





Event generation only if <u>all</u> GTA permit it

3 TA Model of Adaptive DVS Processor

Speed computation in event-driven OPT requires knowledge of remaining execution time and time left to deadline

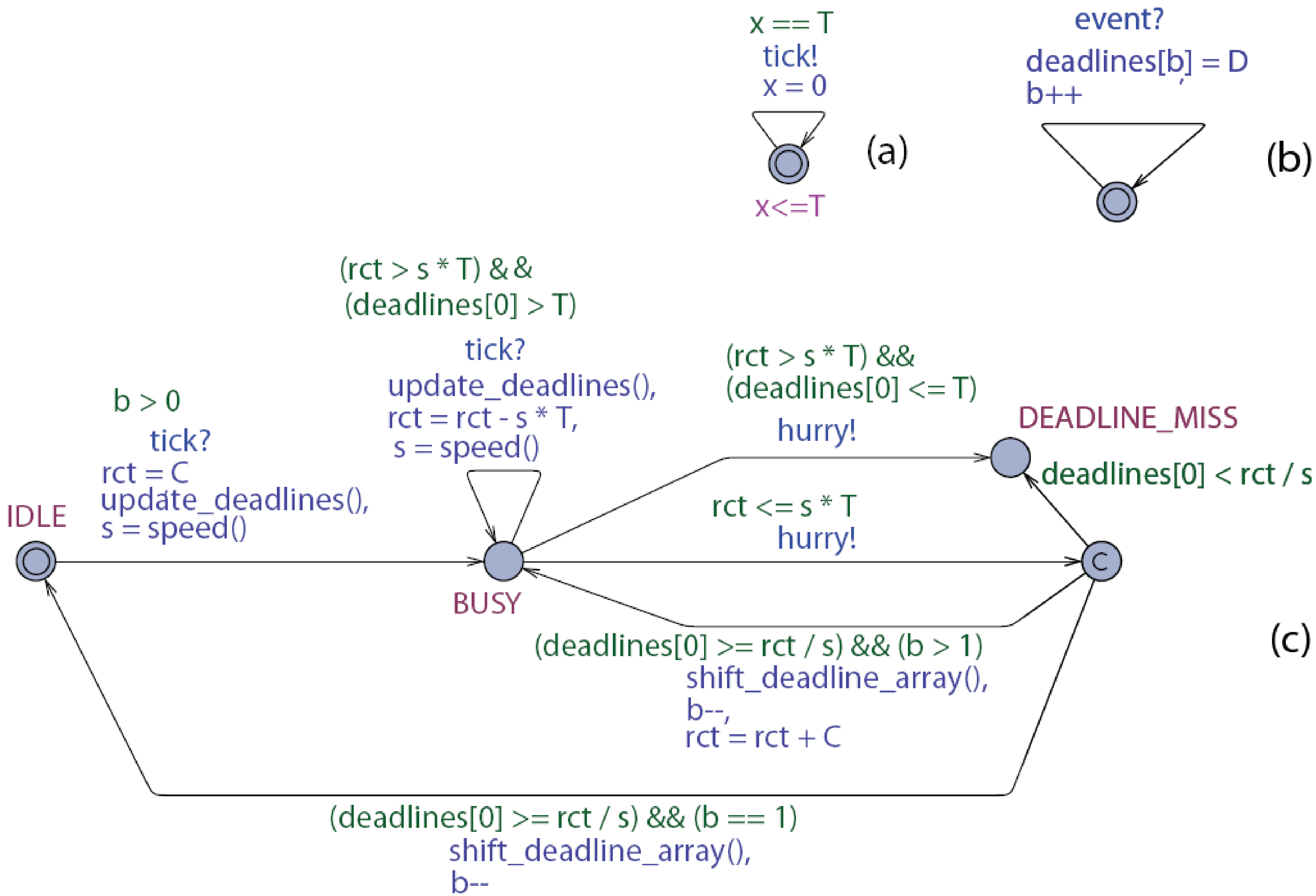
$\Rightarrow$ Exact modeling of OPT is not possible in UPPAAL (because computations on clock variables are not supported)

Finding a conservative TA approximation of event-driven OPT is not trivial!

We devise a formal model for a time-driven variant of OPT

- Based on discrete time (clock ticks with period T)
- Adaptation of event release times: $a' := \lceil \frac{a}{T} \rceil T$
- Adaptation of deadlines: $d' := \lfloor \frac{d}{T} \rfloor T$

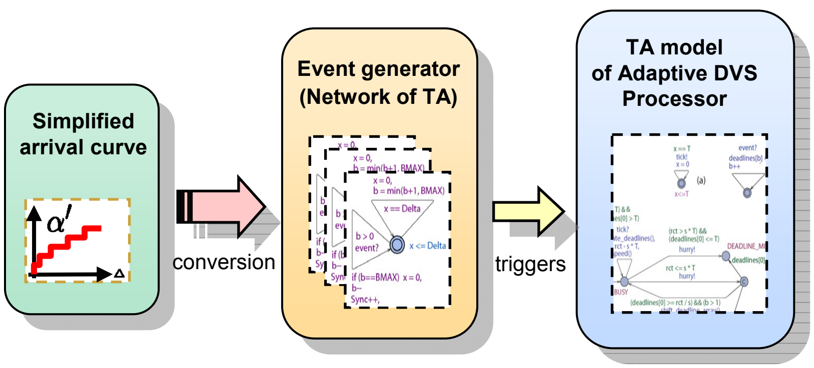
3 TA Model of Adaptive DVS Processor



No additional run-time overhead

Design time

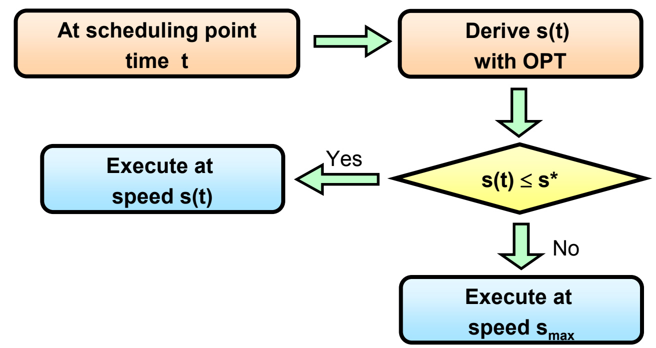
- Parameterization and validation of adaptive DVS scheduler
- Application of (expensive) state-based formal verification



determine threshold speed s*

Run time

- Simple variant of OPT algorithm



use threshold speed s*

Outline

- Motivation
- Background
- Example
- Proposed Adaptive DVS Scheme
- Design and Verifiaction of Adaptive Scheme
- **Experimental Evaluation**
- Conclusions

Experimental Setup

Comparison of static DVS (SD), online DVS (OPT), and adaptive DVS (AD)

Set of 6 periodic event streams with large non-deterministic jitters

Parameters [ms]:

	I	II	III	IV	V	VI
p	198	102	283	239	148	114
J	387	70	269	222	91	13
d	48	45	58	65	78	0
C	30	35	77	69	53	52
D	110	140	310	280	200	120

Considered processor:

Intel XScale

$$s_{max} = 0.5\text{GHz}$$

$$P(s) = 0.04 + \hbar(1.56(\frac{s}{0.5\text{GHz}})^3)\ \text{W}$$

s* for AD is computed with UPPAAL model checker

Results

Maximum speeds for SD and OPT (analytical bounds) [GHz]**:**

	I	II	III	IV	V	VI
s_{SD}	0.44	0.38	0.42	0.4	0.39	0.47
s_{OPT}^{max}	0.513	0.505	0.501	0.506	0.506	0.506

→ OPT violates s_{max}

Maximal threshold speeds s* for AD [GHz]**:**

s^* (T=2ms)	0.38	0.36	0.29	0.39	0.37	0.39

↓ Granularity of discretization in TA model

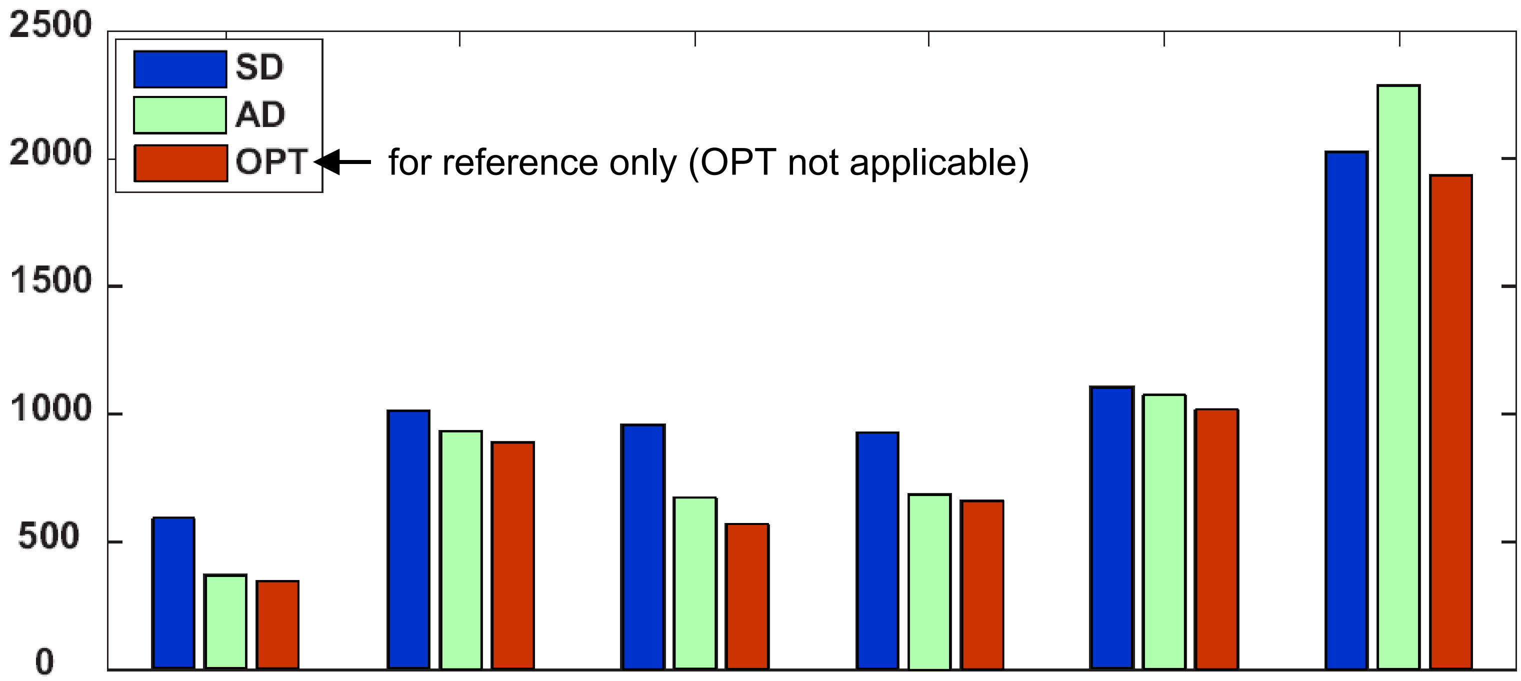
Verification times [s]**:**

(T=2ms)	210	262	16679	2973	459	2

(on a 64 bit Sun Fire X2200 M2 with 8GB RAM)

Results

Average energy consumption for execution of 10 random traces [mJ]**:**



- Adaptive DVS is not much worse than OPT (10% on average)
- Adaptive DVS performs better than Static DVS (22% on average)

Outline

- Motivation
- Background
- Example
- Proposed Adaptive DVS Scheme
- Design and Verification of Adaptive Scheme
- Experimental Evaluation
- **Conclusions**

Conclusions

- New adaptive scheme for DVS scheduling of arbitrary non-deterministic event streams bounded by arrival curves
- Combines advantages of offline and online DVS scheduling
- Verification of state-based scheme by means of timed model checking
- Extension to multiple event streams simple but computationally expensive (state space explosion expected)
- Extension to discrete speeds simple (reduced complexity)
- Method not bound to particular power model (only monotonicity and convexity of energy consumption required)
- Open issue: How to best choose the speed of the pessimistic mode (s_{max} is not necessarily the best option)

Thank you!

Simon Perathoner

perathoner@tik.ee.ethz.ch