

# An Efficient Configuration Methodology for Time-Division Multiplexed Single Resources

Benny Akesson<sup>1</sup>, Anna Minaeva<sup>1</sup>, Premysl Sucha<sup>1</sup>,  
Andrew Nelson<sup>2</sup> and Zdenek Hanzalek<sup>1</sup>

<sup>1</sup>Czech Technical University in Prague

<sup>2</sup>Eindhoven University of Technology



# Trends in Consumer Electronics Systems

- Embedded systems get **increasingly complex**
  - Increasingly complex applications (more functionality)
  - Growing number of applications integrated in a device
  - More applications execute concurrently
  - Requires increased system performance without increasing power
- The resulting complex contemporary platforms
  - are multi-core systems to improve performance/power ratio
  - Resources in the system are **shared** to reduce cost



# Application Requirements

## → Firm real-time requirements (FRT)

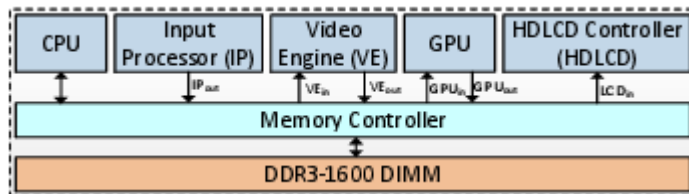
- E.g. Software-defined radio
- Failure to satisfy requirement may violate correctness
- No deadline misses tolerable

## → No real-time requirements (NRT)

- E.g. graphical user interface
- No specified timing requirements, but must be responsive

## → Clients access shared resources on behalf of applications

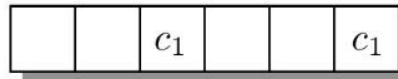
- Require a **minimum bandwidth** and a **maximum latency**



HD video and graphics processing system

# Problem Statement

- Resource sharing results in **interference** between clients
  - Causes resource **contention**
  - Contention is resolved by a **resource arbiter**
  - **Time-Division Multiplexing (TDM)** is commonly used



- Problem is finding a **schedule length** and **slot assignment**
  - that **satisfies bandwidth and latency requirements** of RT clients
  - that **minimizes utilization** to maximize performance of NRT clients
  - in **reasonable computation time**

→The **five main contributions** of this work are:

1. **Latency analysis** for arbitrary slot allocation
2. Formulation of the configuration problem and proof it is **NP-hard**
3. An **optimized ILP formulation** assuming given schedule length
4. **Heuristic algorithm** to choose schedule length
5. **Experimental evaluation** of scalability and trade-offs  
Case study of HD video and graphics processing system



# Presentation Outline

Introduction

**Latency-Rate Servers**

Latency Analysis

Optimized ILP Formulation

Frame-Filtering Heuristic

Experiments

Conclusions

# Latency-Rate Servers

→ Latency-rate servers abstracts service from shared resources

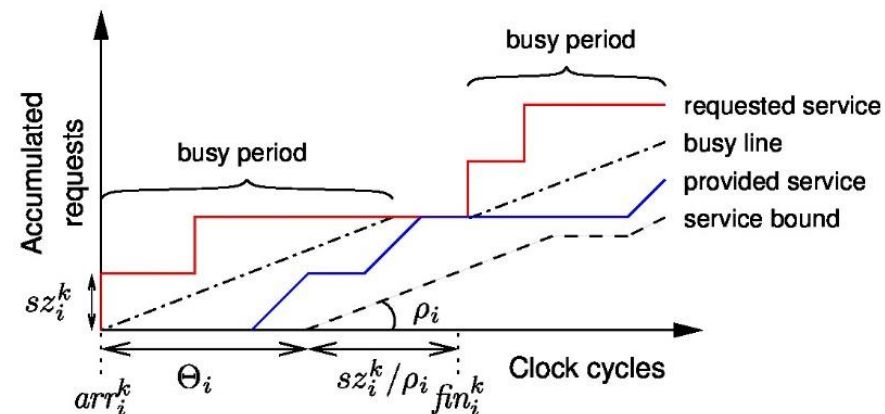
- Client provided **guaranteed rate**,  $\rho$ , after **maximum latency**,  $\Theta$

$$w_i^j \geq \max(0, \rho_i \cdot (j - \Theta_i))$$

- Latency and rate depend on arbiter and its configuration

→ Benefits of latency-rate servers

- Many compatible arbiters
- Works with **sequences of requests**
- Compatible with system-level analysis frameworks



# Time-Division Multiplexing

## →TDM operation

- Periodically repeating schedule (**frame**),  $f$
- Each slot is resource access with bounded ET

## →TDM configuration

- Each client  $i$  allocated  $\phi_i$  slots  $\rho_i = \phi_i / f$
- Exact slots determined by slot assignment policy
- Two simple policies are **continuous** and **equidistant** assignment

$$\Theta_i^{co} = f \cdot (1 - \rho_i) \quad \begin{array}{|c|c|c|c|c|c|} \hline & & & & c_1 & c_1 \\ \hline \end{array} \quad \begin{array}{|c|c|c|c|c|c|} \hline & & c_1 & & & c_1 \\ \hline \end{array} \quad \Theta_i^{eq} = 1/\rho_i - 1$$

$\longleftrightarrow \Theta_1^{co} = 4 \quad \quad \quad \Theta_1^{eq} = 2 \quad \longleftrightarrow$

## →Problem

- Continuous provides worst possible latency
- Equidistant provides best latency, but is not always possible
- **New analysis required** for more complex and irregular assignment



# Presentation Outline

Introduction

Latency-Rate Servers

**Latency Analysis**

Optimized ILP Formulation

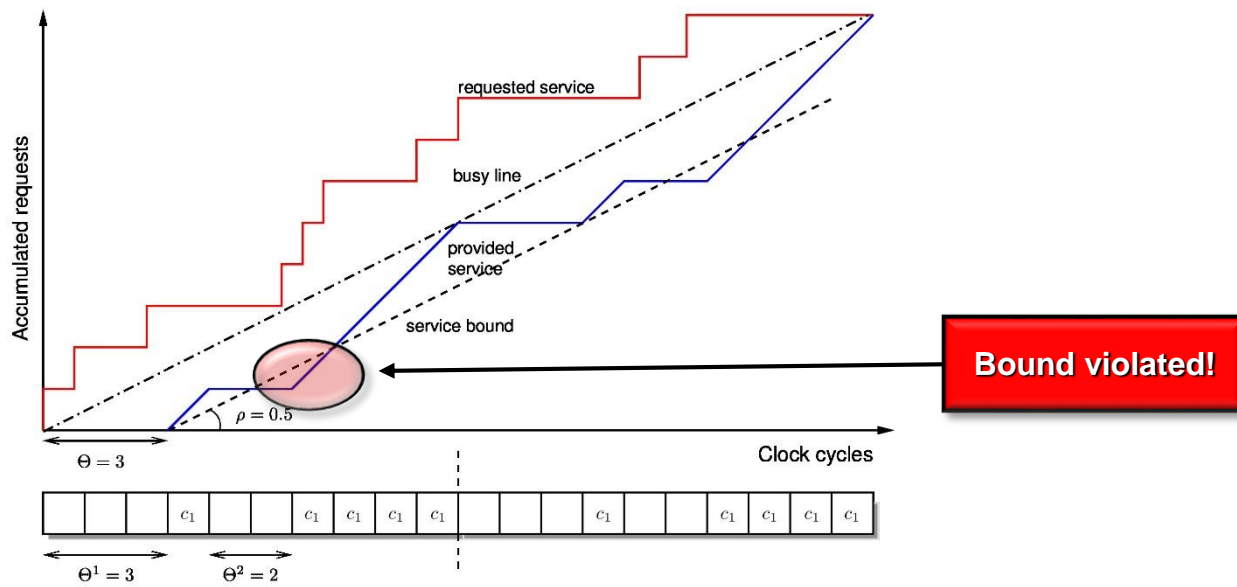
Frame-Filtering Heuristic

Experiments

Conclusions

# Motivational Example

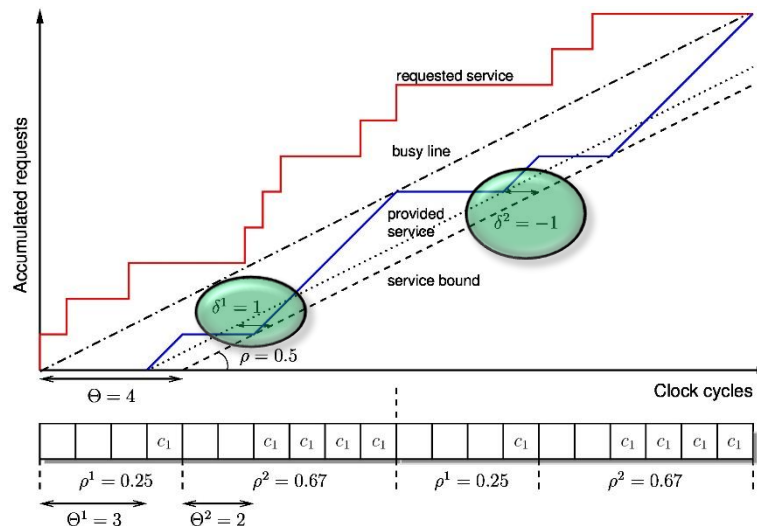
- Computing latency is more difficult than it seems
- Not just largest gap in schedule
  - Must **sustain allocated rate** after latency



Example with frame size 10, 5 allocated slots to  $c_1$ , and largest gap 3

# Sub-Tables and Offsets

- Analysis divides TDM schedule into **sub-tables**
  - Each sub-table has continuous allocation
  - Easy to determine **local latency and rate**
- **Latency offset** computed for each sub-table  $\delta^j = \phi^j + \tilde{\phi}^{j+1} - \phi^j \cdot 1/\rho$ 
  - Ability to sustain rate through idle part of following sub-table
  - Positive offset means longer latency required



# Latency for Arbitrary Allocation

→ Local sub-table analyses combined into **global analysis**

→ We prove that latency is computed according to:

$$\Theta = \max_{j \in [1, N]} \left( \Theta^j + \max \left( 0, \max_{k \in [1, N]} \sum_{l=j}^{j+k-1} \delta^l \right) \right)$$

**For every possible start and end sub-table,  
take maximum of local latency + sum of offsets**

→ Complexity is **quadratic** w.r.t. number of sub-tables

- Max f / 2 sub-tables



# Presentation Outline

Introduction

Latency-Rate Servers

Latency Analysis

**Optimized ILP Formulation**

Frame-Filtering Heuristic

Experiments

Conclusions

# Problem Formulation

- Details of TDM Configuration Problem / Latency-Rate (**TCP/LR**)
  - Determine frame size
  - Determine allocation and slot assignment for all RT clients
  - All latency and rate requirements must be satisfied
  - Total allocated rate must be minimal
- We prove that problem is **NP-hard**
  - Periodic Maintenance Scheduling Problem is a special case

# Basic Model

→ Formulation has **four constraints** and assumes **frame size is given**

1. Each slot is allocated to maximally one client

$$\sum_{c_i \in C} x_i^j \leq 1, \quad j \in F.$$

2. Each client must have enough slots to satisfy rate requirement

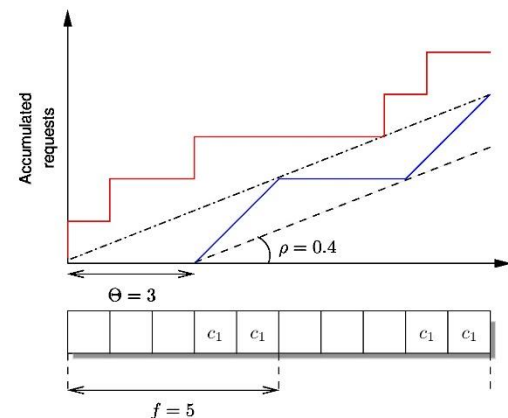
$$\sum_{j=1}^f x_i^j \geq f \cdot \hat{\rho}_i, \quad c_i \in C.$$

3. Worst-case service cannot exceed service provided by TDM table

$$\underline{w}_i^j \leq \sum_{l=k}^{(k+j) \bmod f} x_i^l, \quad k \in F, c_i \in C, j \in F.$$

4. Worst-case service must satisfy latency-rate guarantee

$$\underline{w}_i^j \geq \hat{\rho}_i \cdot (j - \hat{\Theta}_i), \quad j \in F, c_i \in C.$$





# Five Optimizations

- 1. Lower bound on slots** also considers latency requirement
  - Conservatively assumes equidistant allocation
  - Client **latency-dominated** if this is more than rate requirement
- 2. Removing redundant constraints** for latency-dominated clients
  - We prove it is sufficient to check single point on service curve
- 3. Removing rotational symmetry**
  - Give first slot to client with smallest slot requirement
- 4. Value propagation** for different values of frame size
- 5. Checking if allocation feasible after discretization**



# Presentation Outline

Introduction

Latency-Rate Servers

Latency Analysis

Optimized ILP Formulation

**Frame-Filtering Heuristic**

Experiments

Conclusions

# Frame-Filtering Heuristic

- Trying all possible frame sizes might be time-consuming
  - K-heuristic chooses **K candidate sizes** to reduce computation time
  - Implies **trade-off** between computation time and utilization
- Determines **over-allocation** for each candidate
  - Discretized rate – required rate
  - Sort candidates ascending based on total over-allocation
  - Return K first candidates
- Expected behavior
  - Optimal for bandwidth-dominated clients
  - Ok for latency-dominated clients as it prefers large frame sizes



# Presentation Outline

Introduction

Latency-Rate Servers

Latency Analysis

Optimized ILP Formulation

Frame-Filtering Heuristic

**Experiments**

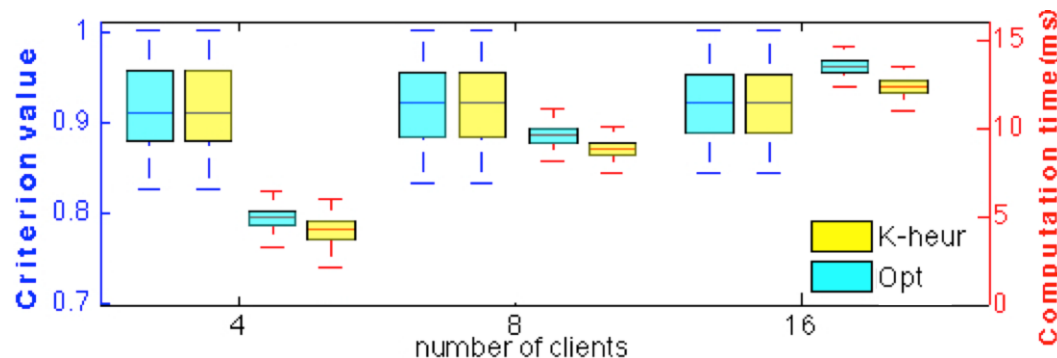
Conclusions

# Experimental Setup

- Two sets of synthetic use-cases with 4, 8 and 16 clients
  - 500 bandwidth-dominated and 500 latency-dominated
  - Frame sizes from  $n$  to  $8n$ , heuristic uses  $K=1$
- Bandwidth-dominated use-cases
  - Total bandwidth requirements in  $[0.8, 0.95]$
  - Relaxed latency requirements
- Latency-dominated use-cases
  - Total bandwidth requirements in  $[0.35, 0.5]$
  - Tighter latency requirements
- CPLEX solver running on 12 Xeon cores with 64 GB memory

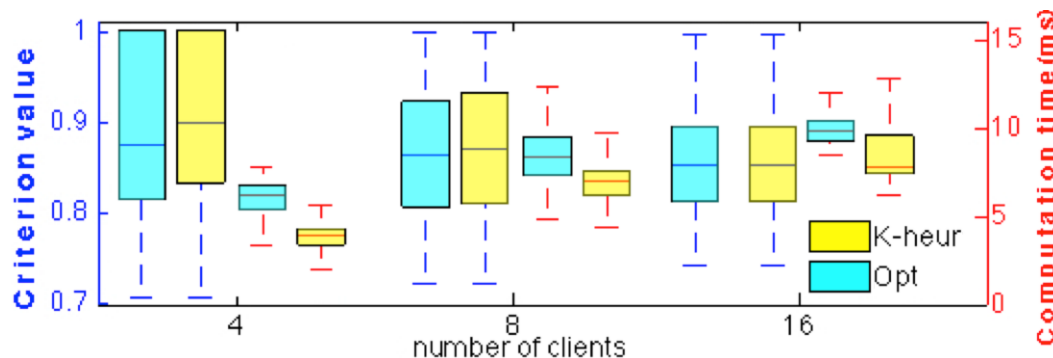
# Bandwidth-Dominated Clients

- Both optimal solution and K-heuristic solve all 1500 use-cases
  - **Computation time increases exponentially** with clients
  - 4 days for optimal solution and 30 hours for K-heuristic
- K-heuristic sub-optimal in 8 use-cases with 4 clients
  - Use-cases close to being latency-dominated
  - Negligible loss of 0.03% in total allocation for these 500 use-cases



# Latency-Dominated Clients

- Optimal solution solves 1500 use-cases and K-heuristic 1496
  - Faster than bandwidth-dominated cases due to optimizations
  - 44 hours for optimal solution, 8 hours for K-heuristic
  - K-heuristic sacrifices 0.5% (median) in total allocation
  - Worse than for bandwidth-dominated clients, which is intuitive
- All experiments repeated with continuous assignment strategy
  - Only succeeds in 452 / 3000 cases, typically with few clients





# Presentation Outline

Introduction

Latency-Rate Servers

Latency Analysis

Optimized ILP Formulation

Frame-Filtering Heuristic

Experiments

**Conclusions**

- This work addresses a **TDM configuration problem**
  - Bandwidth and latency requirements of RT clients must be satisfied
  - Total allocation must be minimized to maximize NRT performance
- We propose
  - A quadratic **latency analysis** for arbitrary slot assignments
  - An **optimized ILP formulation** for the **NP-hard** configuration problem
  - A **heuristic** providing near-optimal results in 28% computation time
- Our approach **outperforms continuous assignment algorithm**
- Demonstrated on HD video and graphics processing system



INVESTMENTS IN EDUCATION DEVELOPMENT

