DATE 13
Design, Automation & Test in Europe
18-22 March, 2013 - Grenoble, France
The European Event for Electronic System Design & Test

Bounding SDRAM Interference: Detailed Analysis vs. Latency Rate Analysis

Hardik Shah
Fortiss GmbH, Germany

Dr. Benny Akesson
CISTER-ISEP Research Centre, Portugal

Prof. Alois Knoll
Technical University Munich, Germany

Agenda

- **Motivation**
- **Background**
- **Detailed Analysis of Credit Controlled Static Priority (CCSP) Arbiter**
- **Optimization of Latency Rate Analysis**
- **Experiments**
- **Conclusion**

Motivation

- **Multi-cores everywhere:**
  - Demanding real-time applications
  - Only multi-cores will be produced in future !!!
- **Interference on shared SDRAM and its effect on the WCET (Worst Case Execution Time) of the hard real-time applications**
  - Shared SDRAM: Cheap – COTS – Complicated
- **Interference analysis:**
  - Detailed analysis
  - Latency rate analysis

Related Work

- **Detailed interference analysis employs precise timing models of shared resource and the arbiter**
  - [1], [2], [9], [10]
- **Latency rate server abstraction [6] is linear lower bound on the service provided by the resource**
  - Shared bus [3], NoC [4] and SRAM/SDRAM [5]
  - Advantages:
    - Resource independent unified modeling
    - Formal performance analysis
- **Comparison of the two analyses in terms of precision is missing**

Contributions

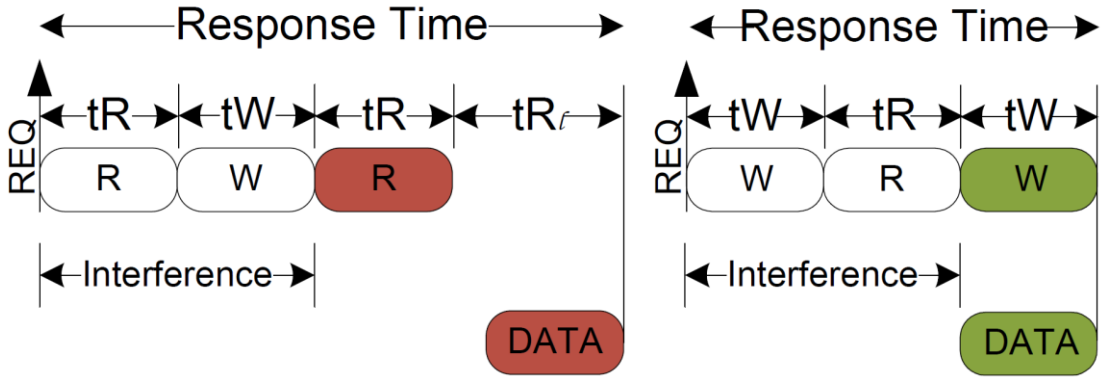
- **Detailed interference analysis of shared SDRAM under the CCSP (Credit Controlled Static Priority) arbitration**
- **Two optimizations to the latency rate analysis based on the detailed analysis**
- **Empirical comparison of the two approaches in terms of produced WCET of applications from CHStone benchmark [8]**

Agenda

- Motivation
- **Background**
- Detailed Analysis of CCSP Arbiter
- Optimization of Latency Rate Analysis
- Experiments
- Conclusion

System Model

- **Multi-core system with shared SDRAM**
- **Closed page policy [1], [2], [5]**
- **Interference as alternating accesses [2]**



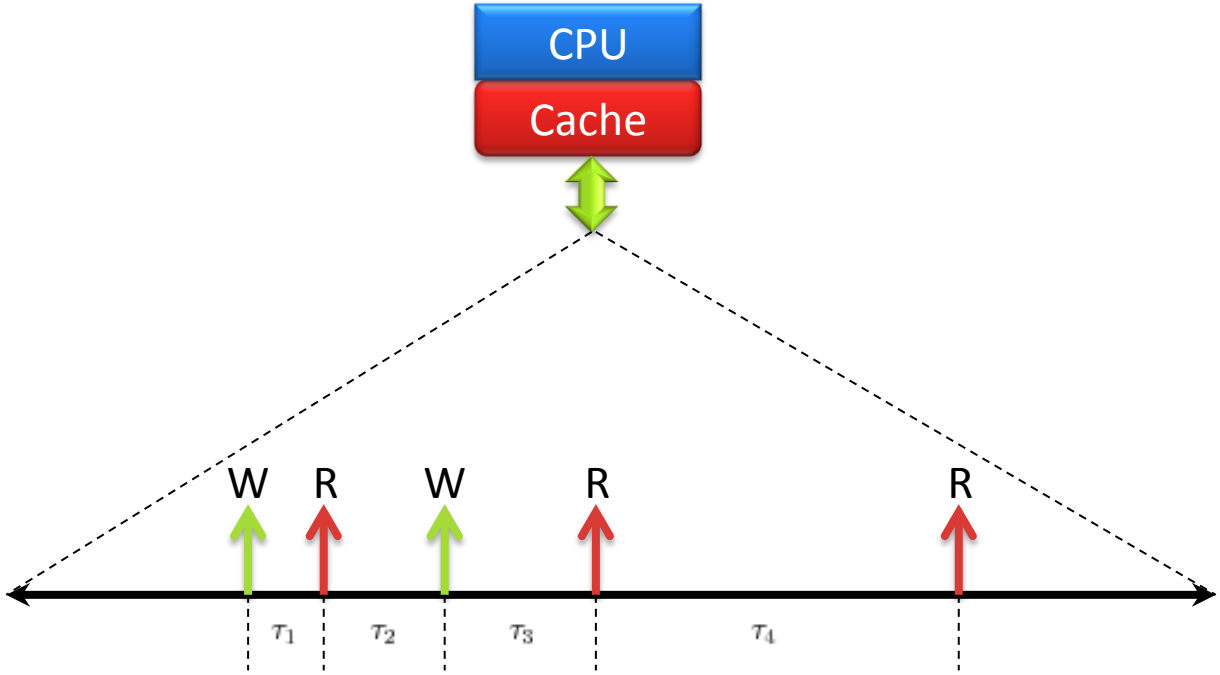
tR = **Worst case read issue time**

tW = **Worst case write issue time**

tR_l = **Worst case read latency**

System Model

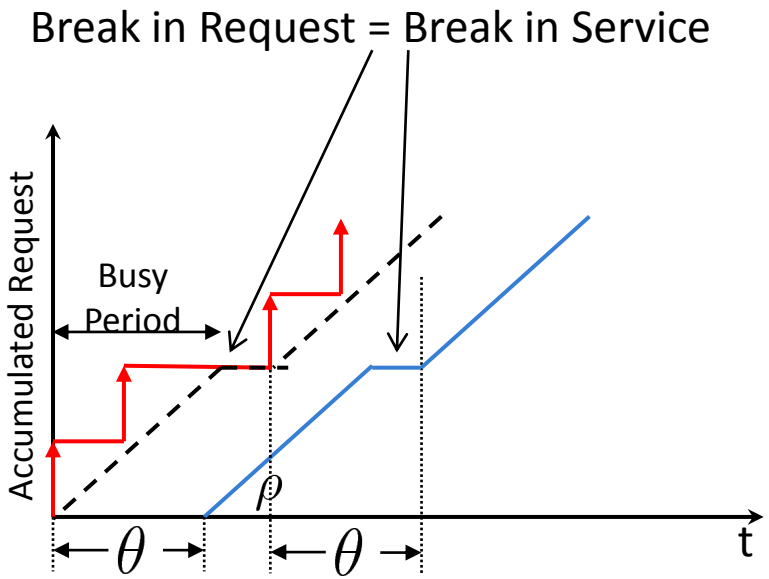
- **Cache-miss trace from a cycle accurate simulator [15]**
- **Without interference**
  - **Interference is added later based on the two analyses**



Latency Rate Analysis

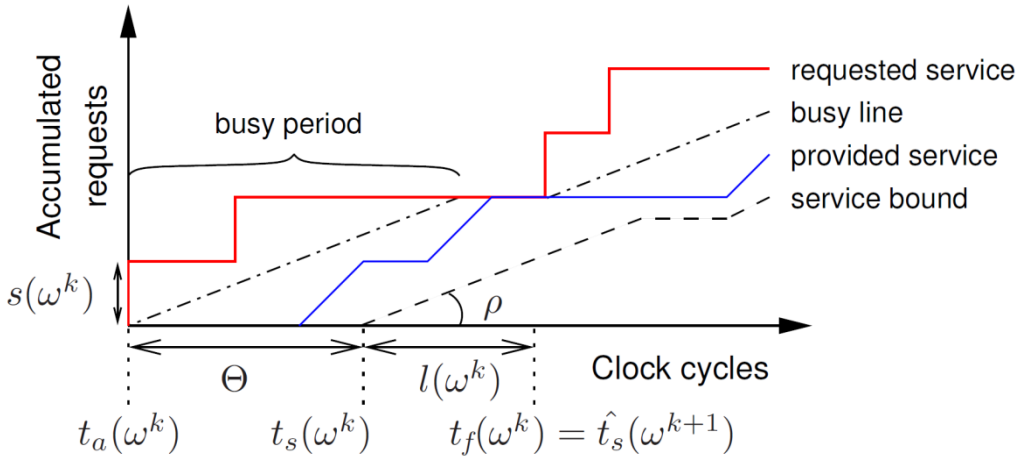
θ = **Maximum latency**

ρ = **Allocated rate**



Latency Rate Analysis

- **Finishing Time**



$$t_f(\omega^k) = \max(t_a(\omega^k) + \Theta, t_f(\omega^{k-1})) + s(\omega^k)/\rho$$

$s(\omega^k)$ = Size of the k^{th} request

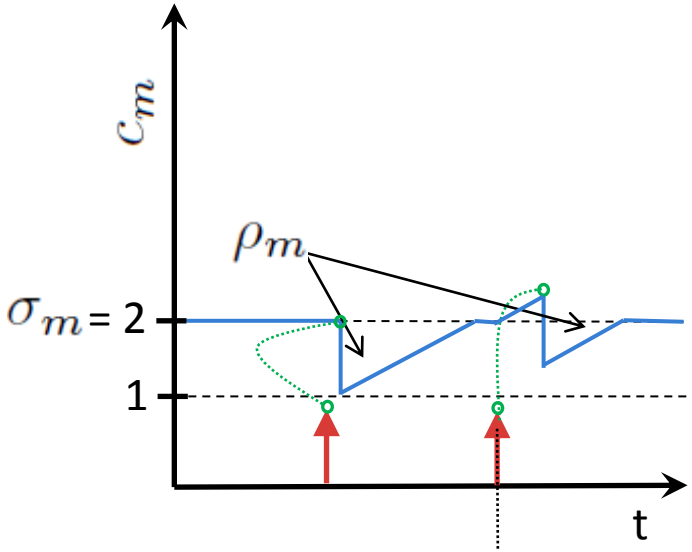
$t_a(\omega^k)$ = Arrival time of the k^{th} request

$t_s(\omega^k)$ = Worst case scheduling time of the k^{th} request

$l(\omega^k)$ = Completion latency of the k^{th} request

Credit Controlled Static Priority Arbiter

- **CCSP**
  - Each master is assigned
    - Initial credit = σ_m
    - Allocated rate = ρ_m
    - Static priority



$$c_m(0) = \sigma_m$$

$$c_m(t+1) = \begin{cases} c_m(t) + \rho_m - 1 & \gamma(t) = m \\ c_m(t) + \rho_m & \gamma(t) \neq m \wedge b_m > 0 \\ \min(c_m(t) + \rho_m, \sigma_m) & \gamma(t) \neq m \wedge b_m = 0 \end{cases}$$

CCSP Arbiter

- **Due to the static priority, the scheduling latency of an access depends on the available credits of higher priority masters and their allocated rate**

$$\Theta_m = \frac{\sum_{\forall m_j \in M_m^+} \sigma_{m_j}}{1 - \sum_{\forall m_j \in M_m^+} \rho_{m_j}}$$

M_m^+ - Set of Higher Priority Masters

  - High allocation to the higher priority masters leads to infinite latency

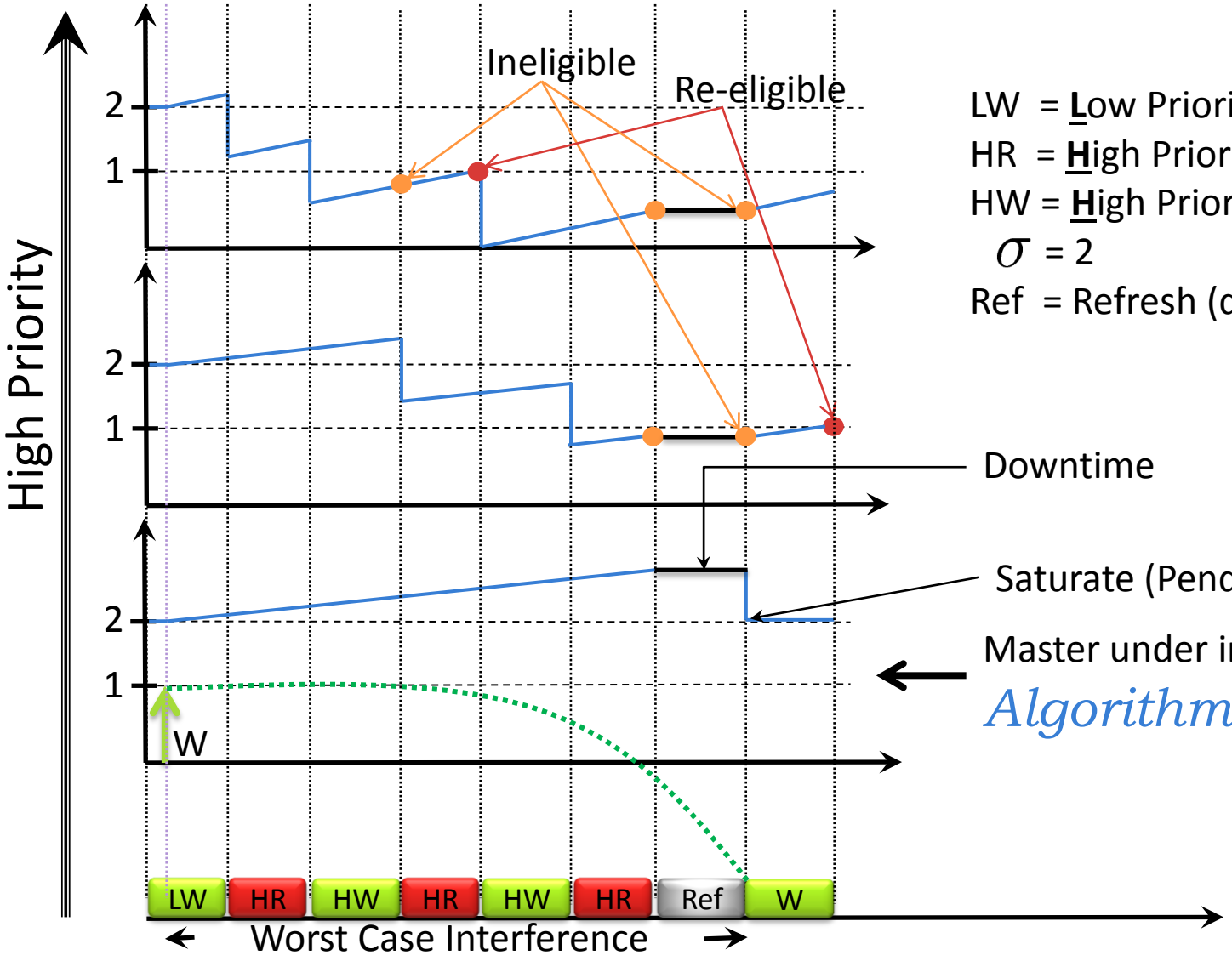
Agenda

- Motivation
- Background
- **Detailed Analysis of CCSP Arbiter**
- Optimization of Latency Rate Analysis
- Experiments
- Conclusion

Detailed Worst Case Interference Analysis

- **Intuitions for worst case interference analysis of master m**
  1. Interfering accesses and the access from m form alternating sequence of accesses towards SDRAM
  2. All other masters use their credits only to interfere with m
     - When m is not requesting, other masters also do not request and accumulate as many credits as possible
     - All high priority masters request together with m
  3. One lower priority master requests an access one clock cycle before m requests an access
  4. One refresh interferes at every tREFI clock cycles

Detailed Analysis



LW = **L**ow Priority **W**rite
HR = **H**igh Priority **R**ead
HW = **H**igh Priority **W**rite
σ = 2
Ref = Refresh (downtime @ tREFI)

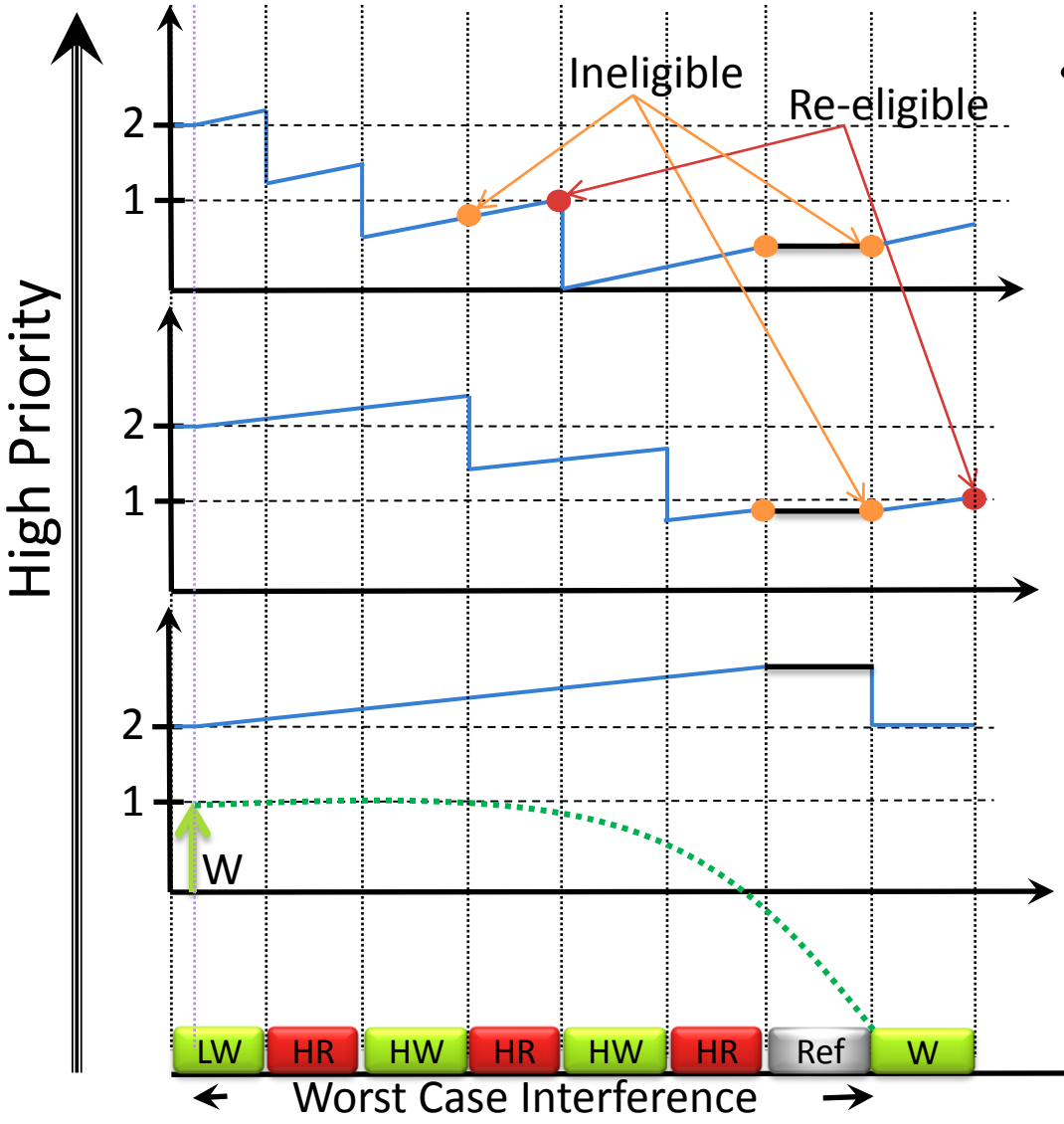
Downtime

Saturate (PendingRequest = 0)

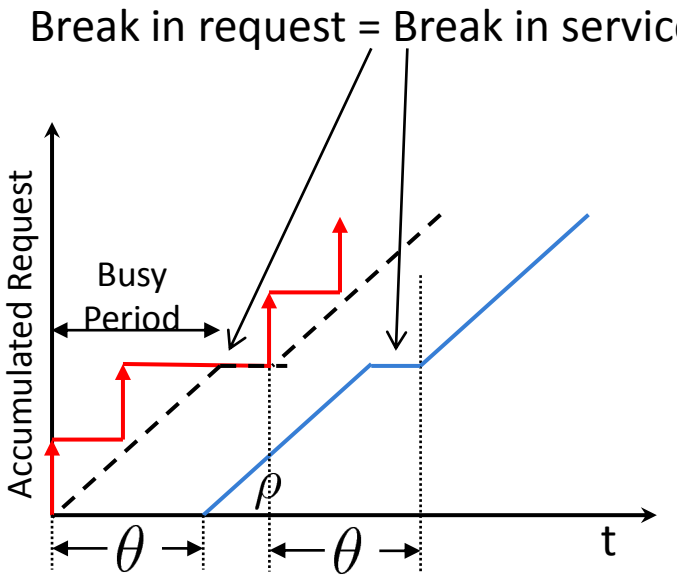
Master under investigation ***m***

Algorithm in the paper

Detailed Analysis



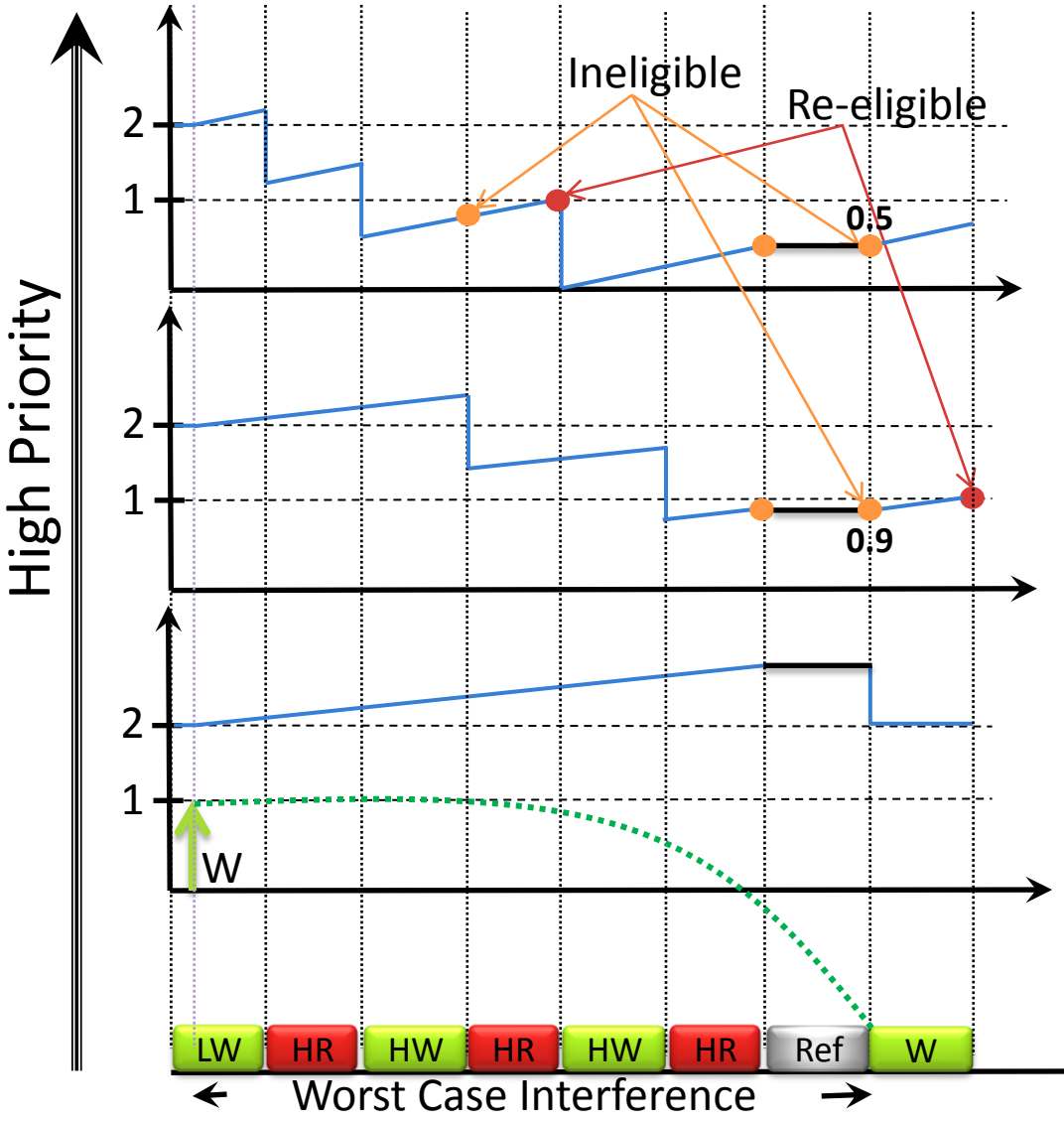
- For next access of m the worst case latency is lower, provided that m requests accesses fast enough to refrain high priority masters accumulate high credits



Agenda

- Motivation
- Background
- Detailed Analysis of CCSP Arbiter
- **Optimization of Latency Rate Analysis**
- Experiments
- Conclusion

Optimized Bound on Latency



Latency rate analysis considers interference from high priority masters after summing-up their credits **~ 1.4**

$$\Theta_m = \frac{\sum_{\forall m_j \in M_m^+} \sigma_{m_j}}{1 - \sum_{\forall m_j \in M_m^+} \rho_{m_j}}$$

During execution, only masters with at least one credit can interfere

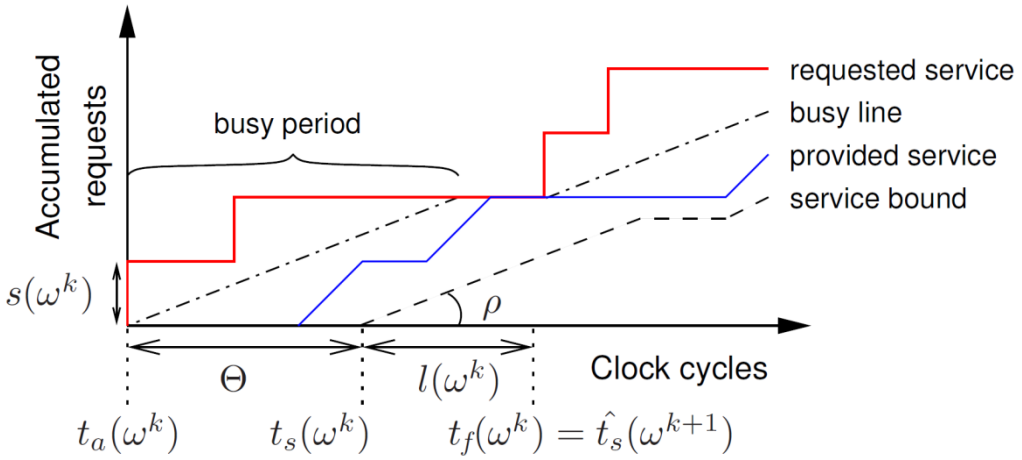
$$\Theta_m^0 = 0,\ \Theta_m^1 = \sum_{\forall m_j \in M_m^+} \sigma_{m_j}$$

$$\Theta_m^k = \Theta_m^{k-1} + \sum_{\forall m_j \in M_m^+} \left\lfloor (\Theta_m^{k-1} - \Theta_m^{k-2}) \times \rho_{m_j} \right\rfloor$$

Improves precision of analysis for low priority masters

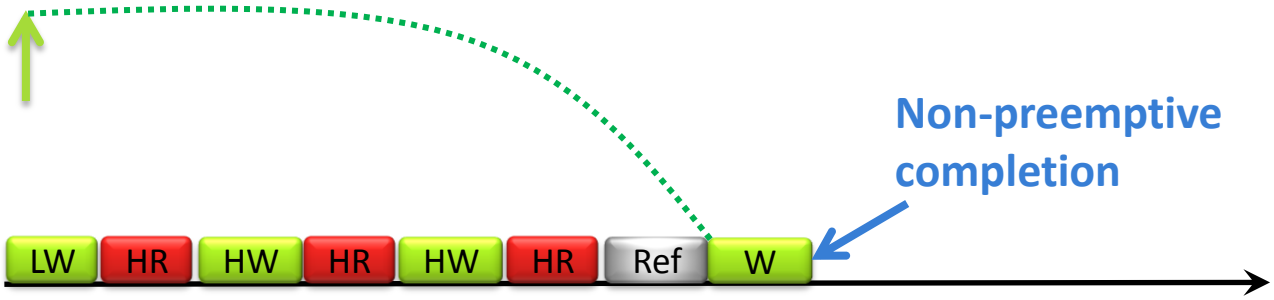
Optimized Finishing Time

- **Latency rate analysis:**



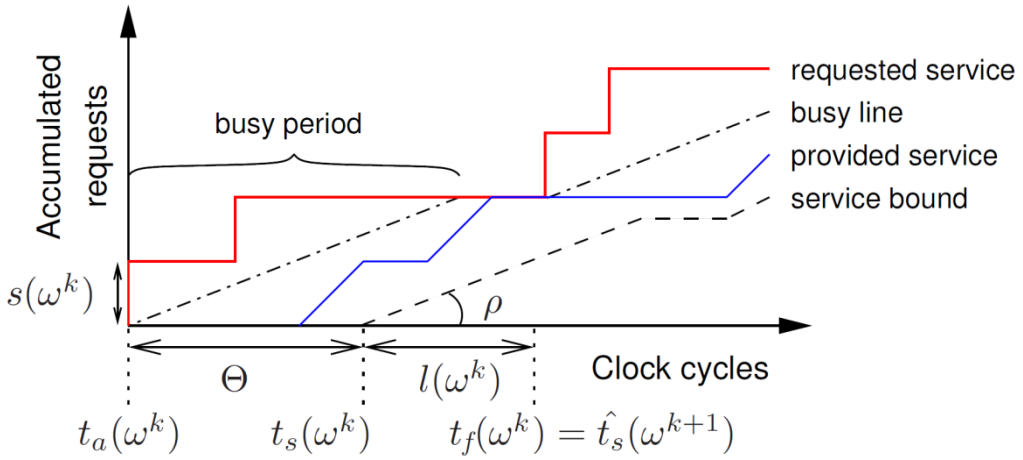
$$t_f(\omega^k) = \max(t_a(\omega^k) + \Theta, t_f(\omega^{k-1})) + s(\omega^k)/\rho$$

Fluid model

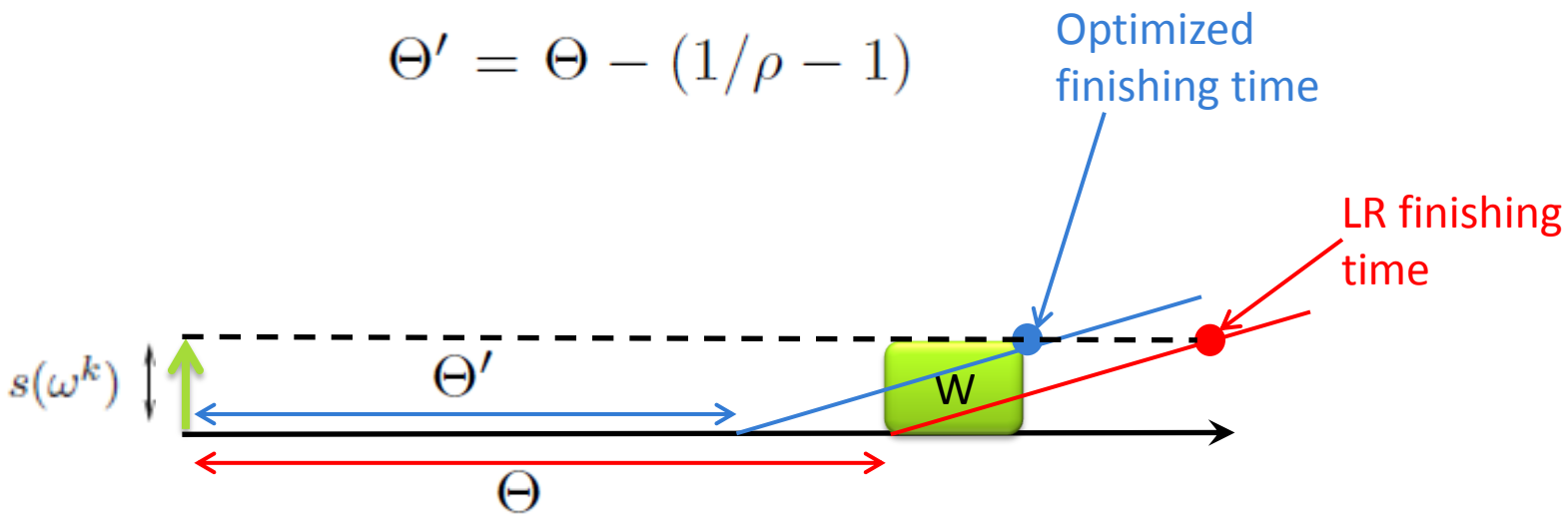


Optimized Finishing Time

- **Latency rate analysis:**

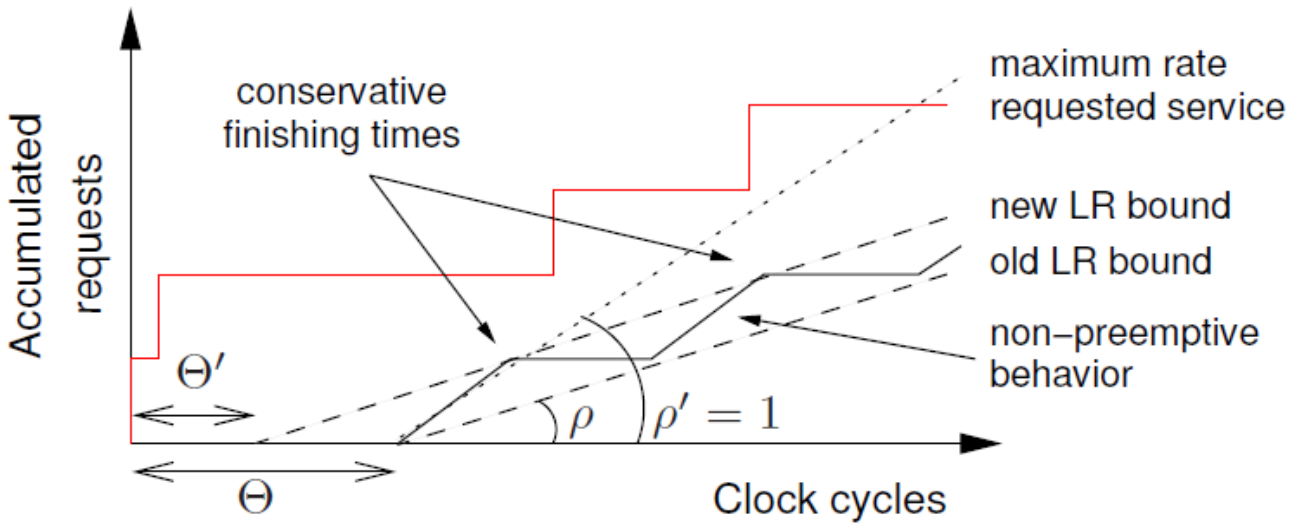


$$\Theta' = \Theta - (1/\rho - 1)$$



Optimized Finishing Time

- **New LR bound with non-preemptive behavior**



$$\Theta' = \Theta - (1/\rho - 1)$$

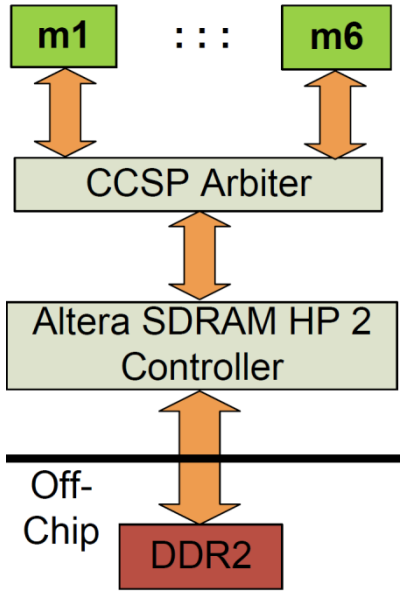
Improves precision of analysis for all masters

Agenda

- Motivation
- Background
- Detailed Analysis of CCSP Arbiter
- Optimization of Latency Rate Analysis
- **Experiments**
- Conclusion

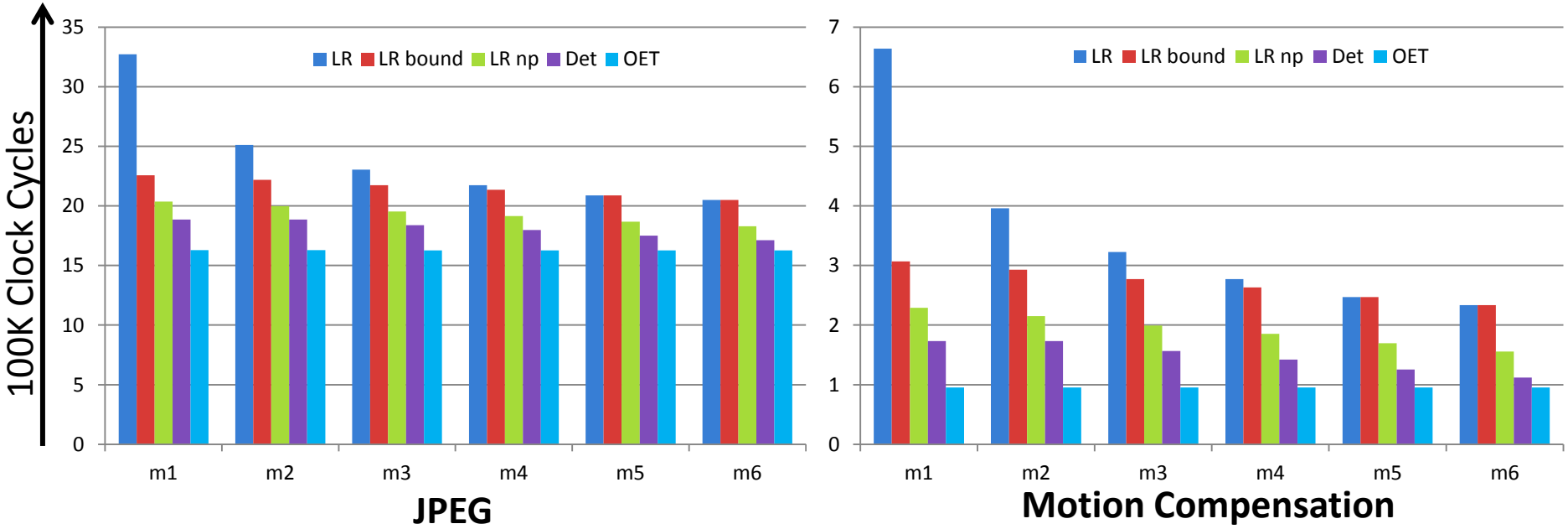
Test Setup

- **Altera cyclone III FPGA + Micron 667 DDR2**
- **CHStone benchmark cache-miss traces**
  - JPEG (least memory intensive)
  - Motion compensation(most memory intensive)
- **Same application – same path on six hardware trace players executing on the shared DDR2**
- **M6 -> highest priority, M1 -> lowest priority**



Experiment 1

- **Equal allocations** ρ **= 1/6**



LR: Default latency rate analysis

LR bound: LR optimization - Bounded latency

LR np: LR optimization - Non-preemptive service + **Bound Latency**

Det: Detailed analysis

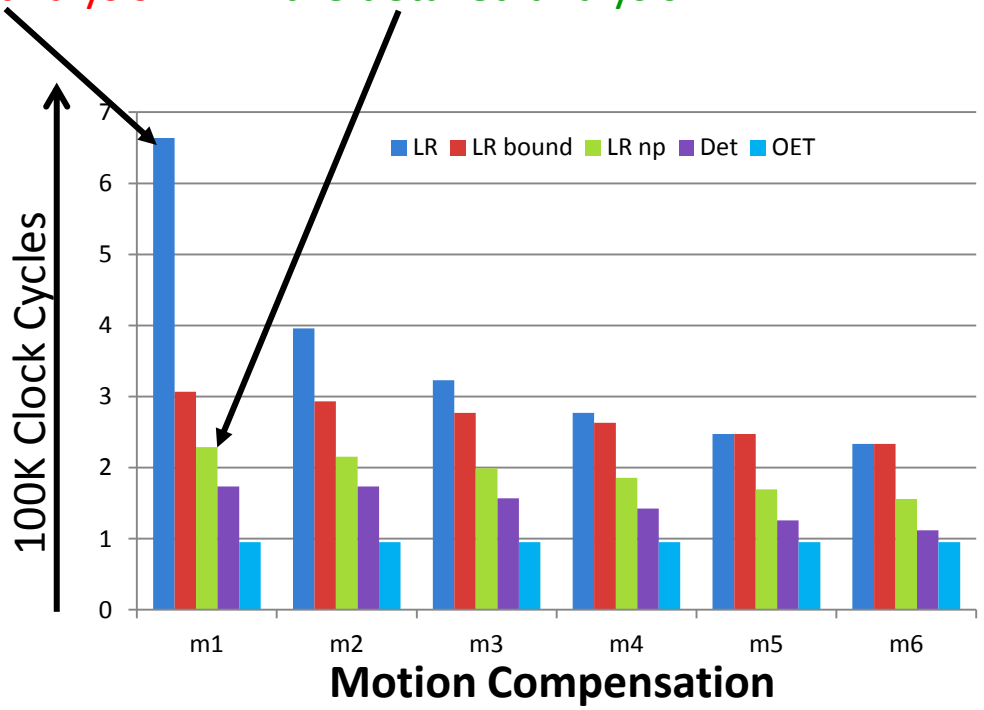
Oet: Observed Execution Time

Experiment 1

- **Equal allocations** ρ **= 1/6**

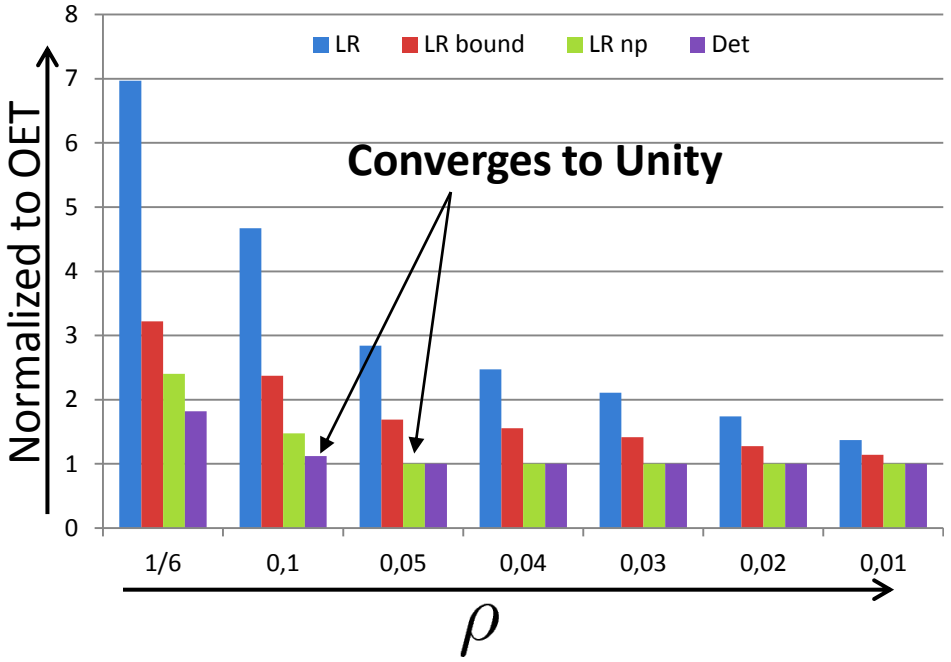
3.8 Times larger than the detailed analysis

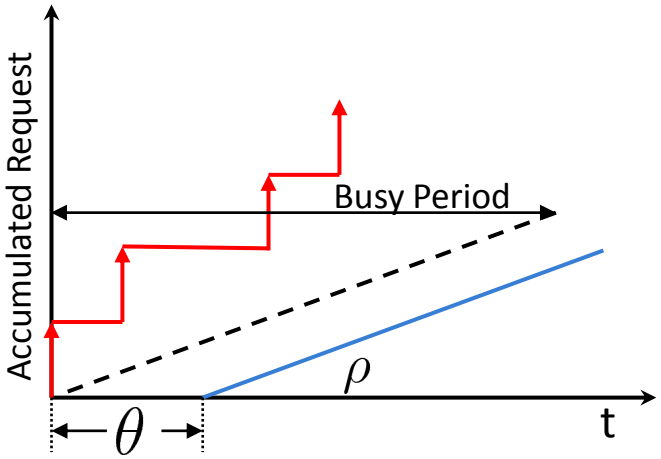
1.4 Times larger than the detailed analysis



Experiment 2

- **Reduced allocation**
  - Reduced allocated rate of the lowest priority master
  - Improved precision





Agenda

- Motivation
- Background
- Detailed Analysis of CCSP Arbiter
- Optimization of Latency Rate Analysis
- Experiments
- **Conclusion**

Conclusion

- **Detailed worst case interference analysis of SDRAM shared under the CCSP arbitration**
- **Two optimization of native latency rate analysis based on the detailed analysis**
  - Bounded latency helpful to low priority masters
  - Non-preemptive scheduling helpful to all masters
- **Comparison of both analyses in terms of WCET produced by them of real application**
  - Precision of LR analysis depends on the master's ability to keep the server busy