Hybrid Co-scheduling Optimizations for Concurrent Applications in Virtualized Environments

Yulong Yu

School of Software

Dalian University of Technology

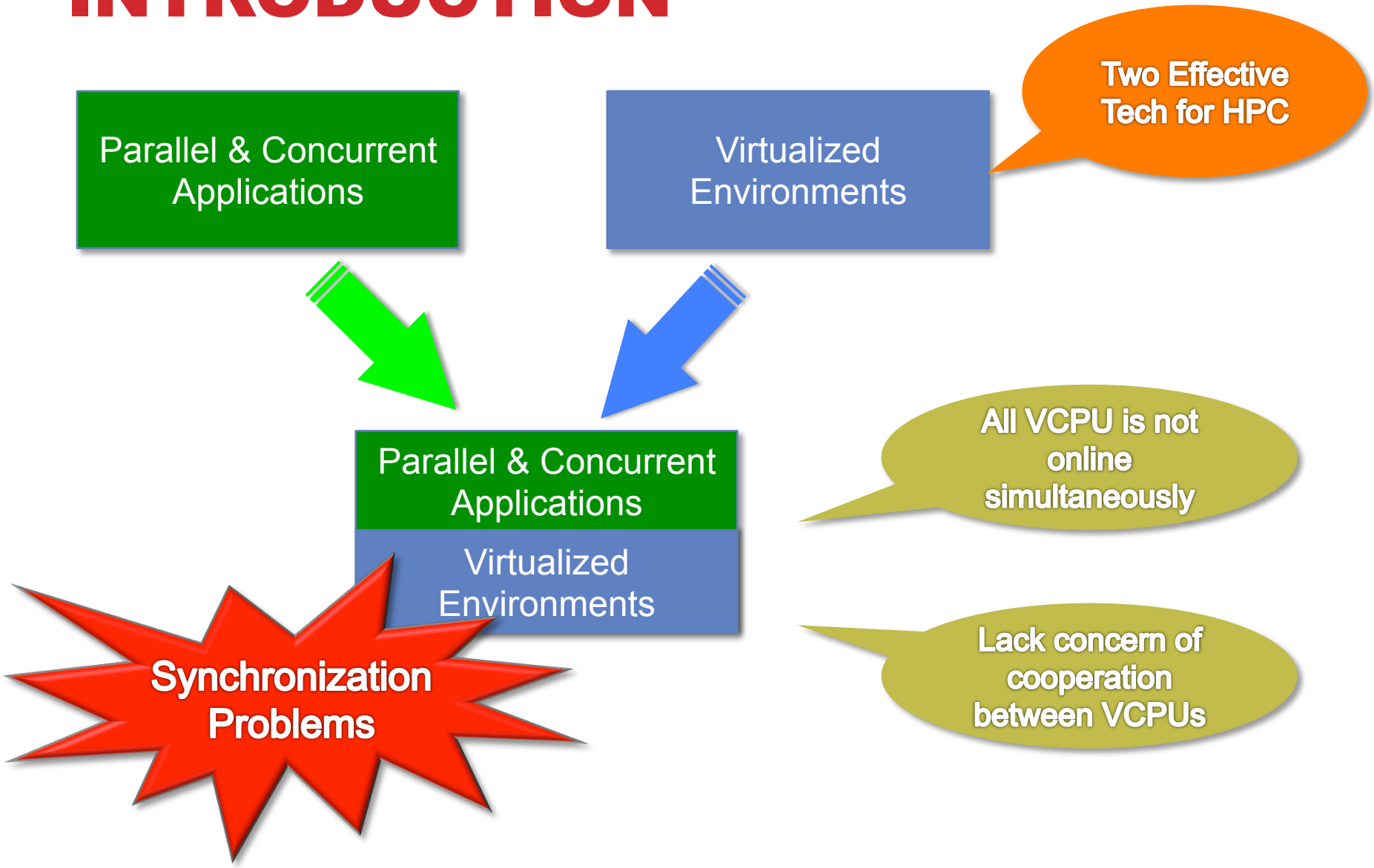
The 6th International Conference on Networking, Architecture, and Storage (NAS), July 28-30, 2011, Dalian, China

HPC&IP

OUTLINES

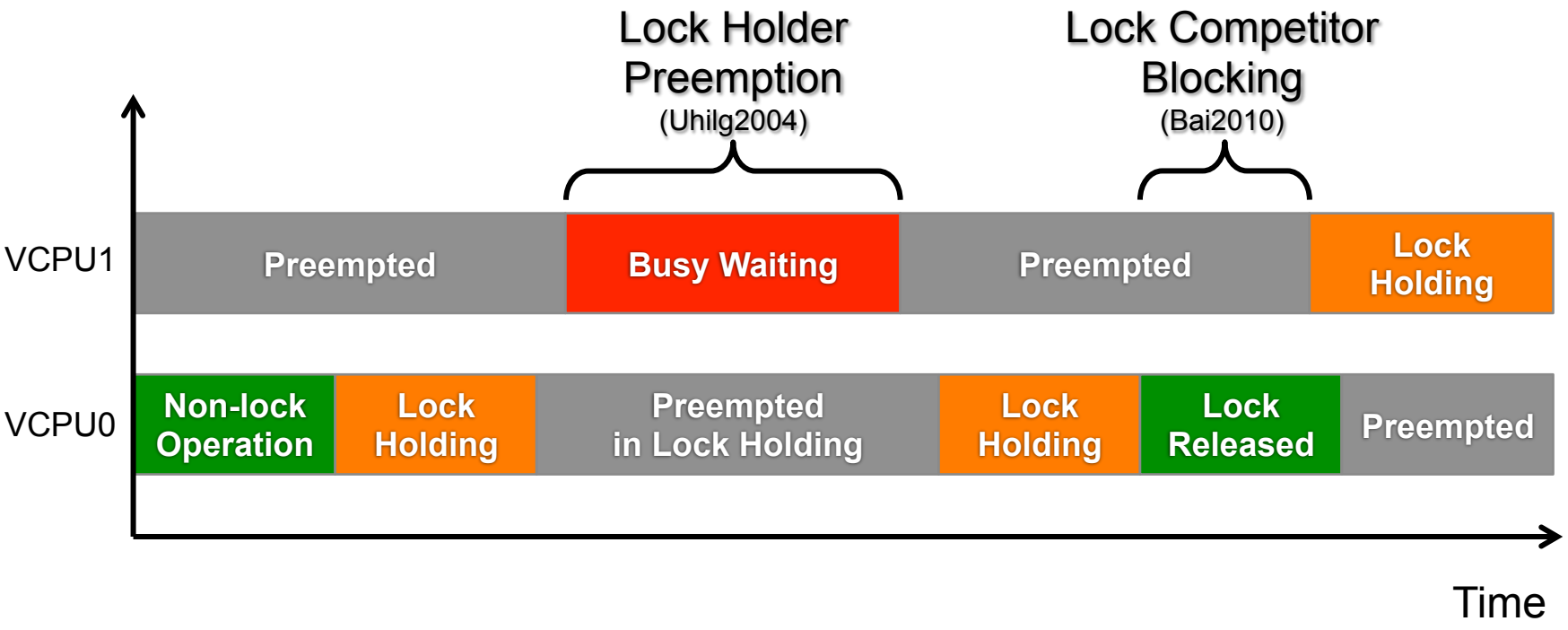
- **Synchronization Problems for Concurrent Applications in VE**
- **Co-scheduling in VE and Its Problems**
- **Two schemes we proposed:**
  - Partial Co-scheduling
  - Boost Co-scheduling
- **Comparison between two schemes**
- **Experiments and Measurement Results**
- **Conclusions and Future Work**

INTRODUCTION



INTRODUCTION

- **Synchronization Problems for Concurrent Applications in Virtualized Environments**



INTRODUCTION

- **Current existing work for Synchronization Problems (Intrusive & Non-intrusive Methods)**
  - Intrusive Methods
    (Actions based on the sematic detection)
    - Lock-aware Delay Preemption (Uhlig2004)
    - Spin Yield (Jiang2009)
    - Active Waiting Prevention (Friebel2008)
  - Non-intrusive Methods
    (Actions to keep the prerequisite in native environments)
    - Co-Scheduling (Weng2009)
    - Gang-Scheduling (Feitelson1994)
  - In *Intrusive Methods*, Detection Algorithms or Modified Guest OS is necessary to discover the co-operations between VCPUs, which brings more complexity than *Non-intrusive Methods*.

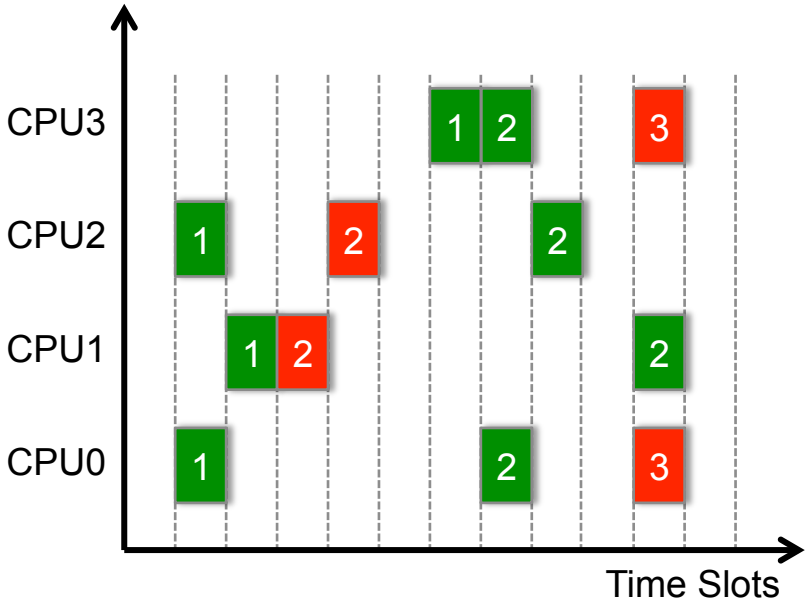
CO-SCHEDULING IN VE

- **Definition**
  - All the VCPUs that belong to a VM are scheduled simultaneously.
- **Benefits**
  - Keeping the simultaneous online prerequisite in native environments.
  - No semantic detection or modified guest OS requirement
  - Orthogonal to underlying scheduler
- **Current co-scheduling solutions**
  - Hybrid Co-scheduling (Weng2009)
  - Co-de-scheduling (VMWare2008, Jiang2009)
  - Task-aware Co-scheduling (Xu2009, Bai2010)
  - Approximate Co-scheduling (Jiang2009)

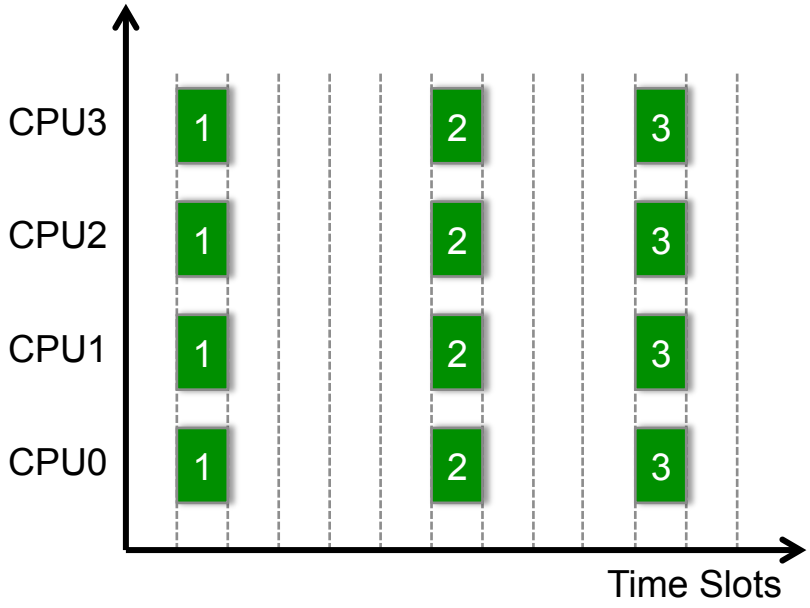
CO-SCHEDULING IN VE

- **Scenarios without or with Co-scheduling**

C. Weng, Z. Wang, M. Li, et al. The hybrid scheduling framework for virtual machine system, in VEE'09, pp. 111-120



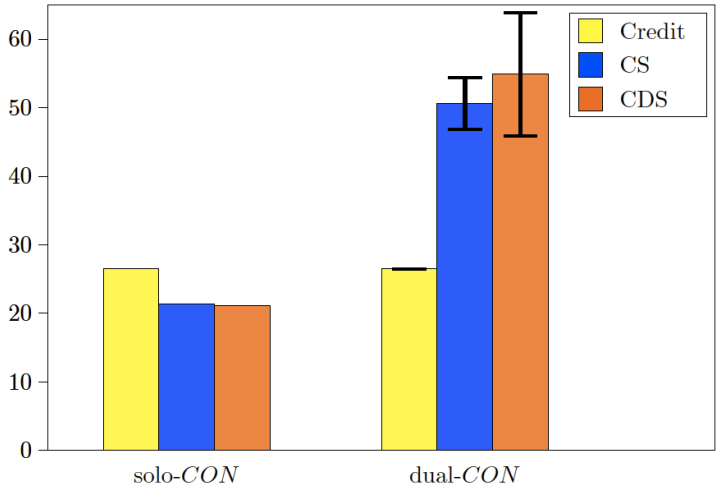
Non-co-scheduling Scenario



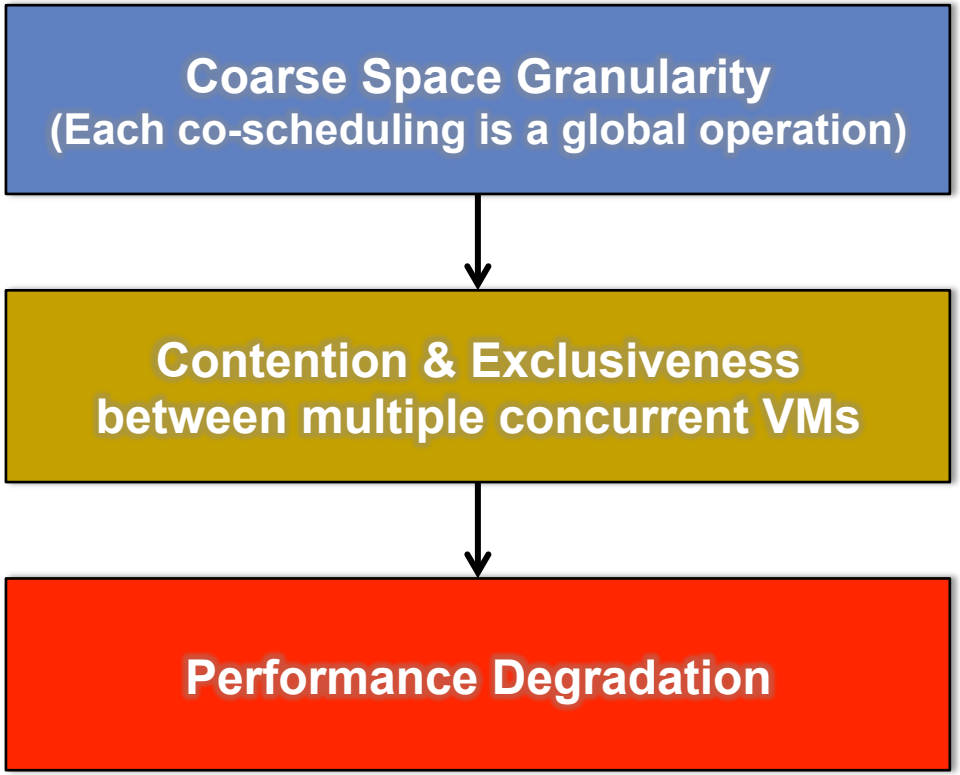
Co-scheduling Scenario

CO-SCHEDULING IN VE

- **Problems in current Hybrid Co-scheduling**
  - When multiple concurrent VMs co-exists in system, Hybrid Co-scheduling performance degrades seriously.

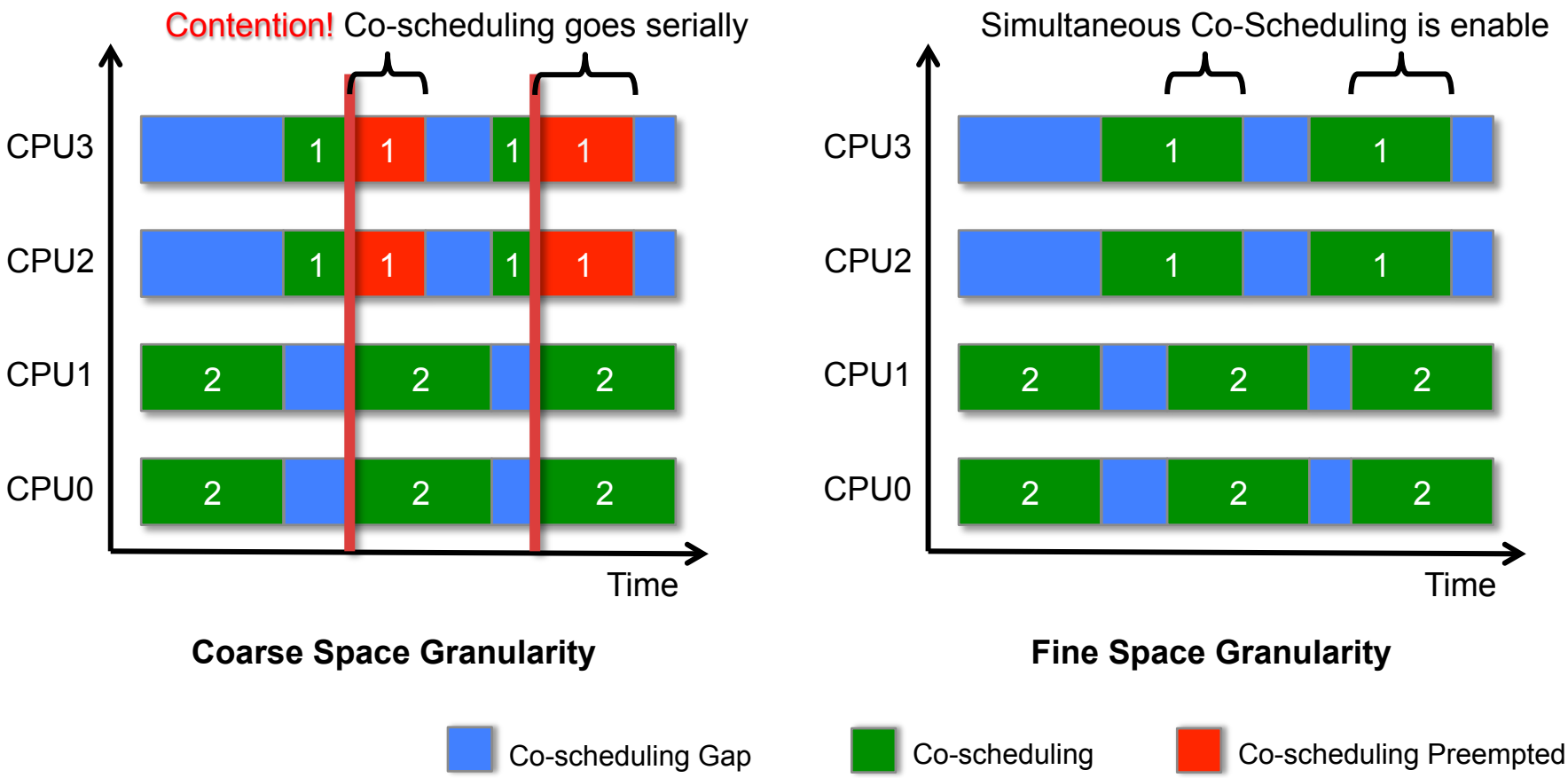


Execution time of LU with different scheduling schemes



CO-SCHEDULING IN VE

- **Coarse & Fine Space Granularity in Co-scheduling**

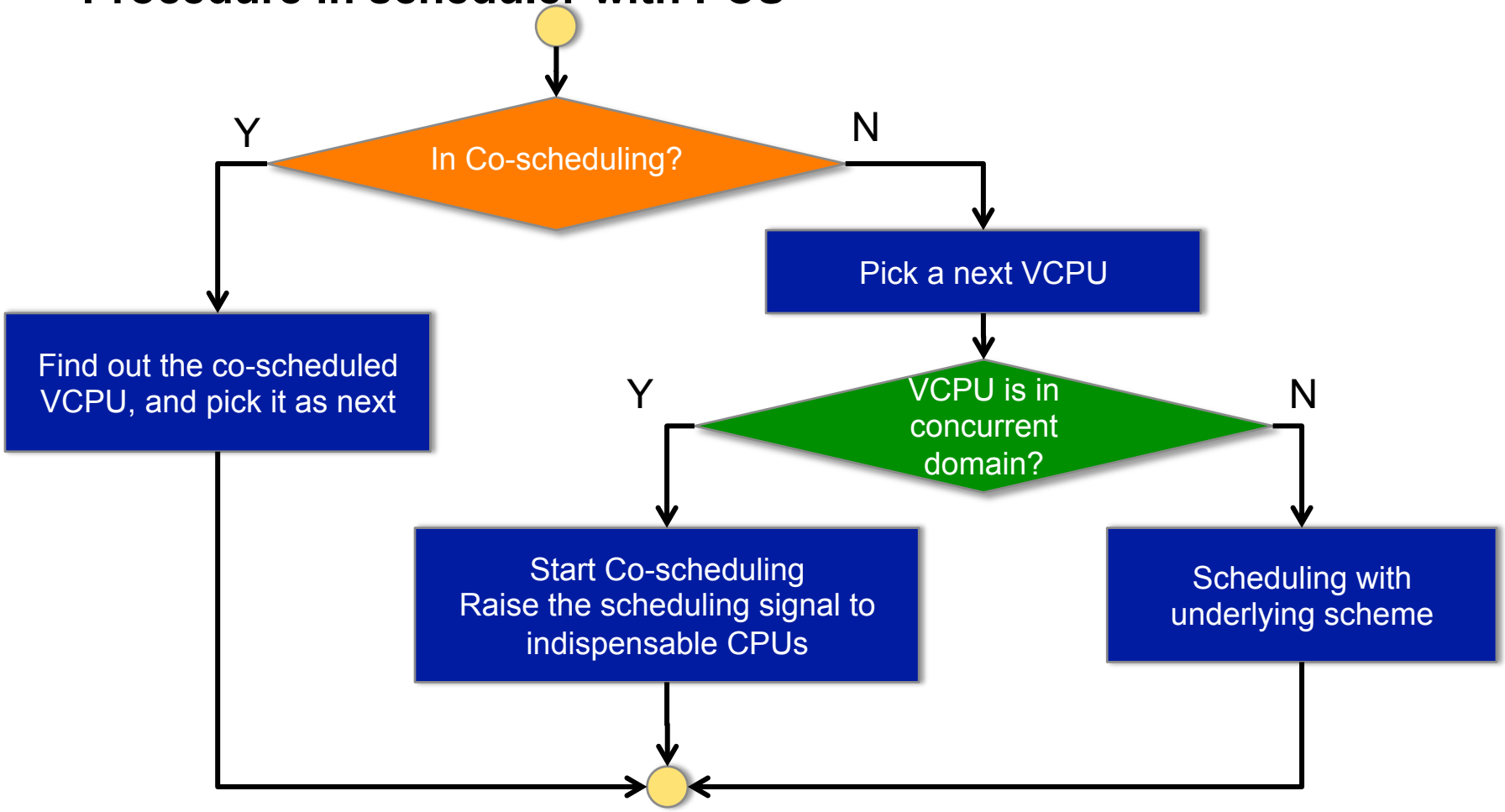


PARTIAL CO-SCHEDULING (PCS)

- **General Idea**
  - Sending the co-scheduling signal to indispensable CPUs instead of to all online CPUs
- **Implementation Key Points**
  - Recording the co-scheduling state for each online CPU, not just for the whole system
  - Recording the VCPU distribution throughout online CPUs for each VM

PARTIAL CO-SCHEDULING (PCS)

- **Procedure in scheduler with PCS**



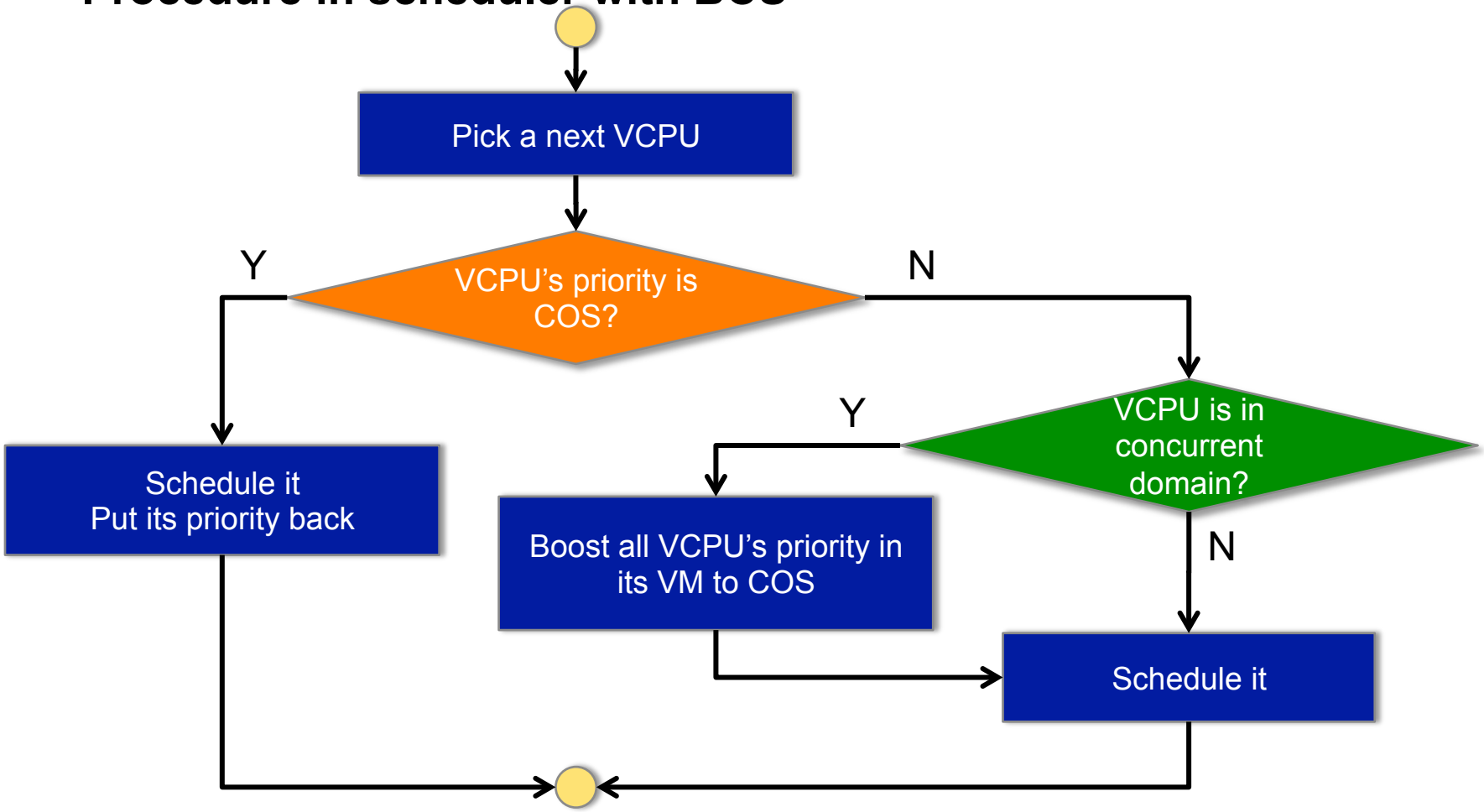
BOOST CO-SCHEDULING (BCS)

- **General Idea**
  - Boost the priorities of co-scheduled VCPUs to induce the underlying scheduler to pick the appropriate VCPUs.

- **Implement Key Points**
  - Introduce a new highest priority into the scheduler -- COS
  - Boost the priorities of co-scheduled VCPUs temporarily

BOOST CO-SCHEDULING (BCS)

- **Procedure in scheduler with BCS**



COMPARISON BETWEEN PCS & BCS

PARTIAL CO-SCHEDULING

- Precise time edge alignment
- Complex implementation, More codes than hybrid co-scheduling
- Perform well and stable in all kinds of concurrency

BOOST CO-SCHEDULING

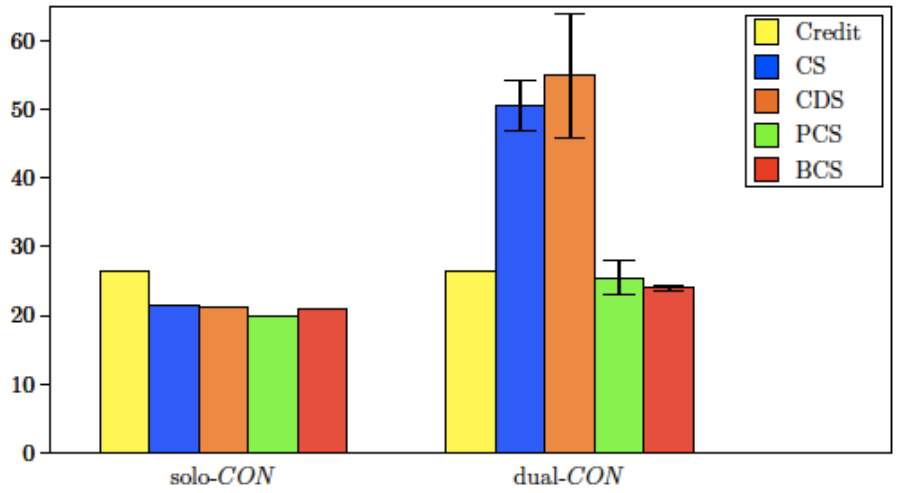
- Imprecise time edge alignment
- Easy implementation, Less code, Better reliability
- Fit most condition except cross domain concurrency

EXPERIMENTS

- **Test bed**
  - Hardware:
    - CPU: quad-core Core i5,
    - Mem: 4GB DDR3
  - Software:
    - Xen 4.0.1 + Ubuntu 10.04 Server
  - Virtual Machine:
    - Dual-core CPU + 394MB Mem
    - CentOS 5.5
- **Benchmarks**
  - SPLASH2 LU kernel
    - P=2, N=4096, B=16
  - NPB: six benchmarks selected
    - BT, CG, EP, FT, LU, MG (Class A & B)

EXPERIMENTS

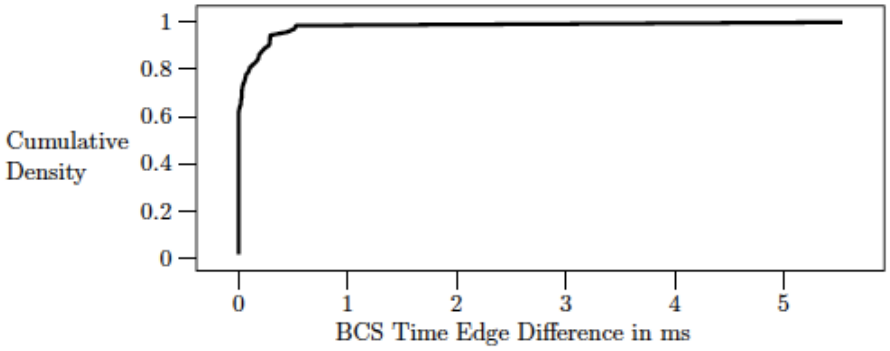
- **LU Experiment**
  - Execution time
  - Co-scheduling frequency
  - Time edge difference in BCS



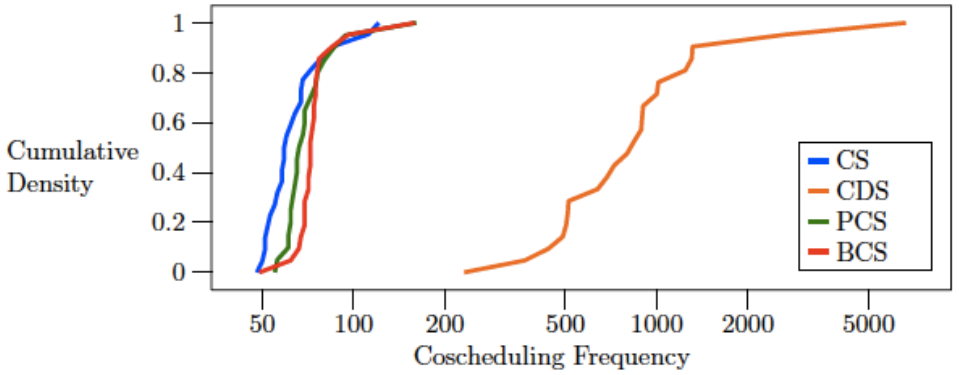
Execution Time (sec)

EXPERIMENTS

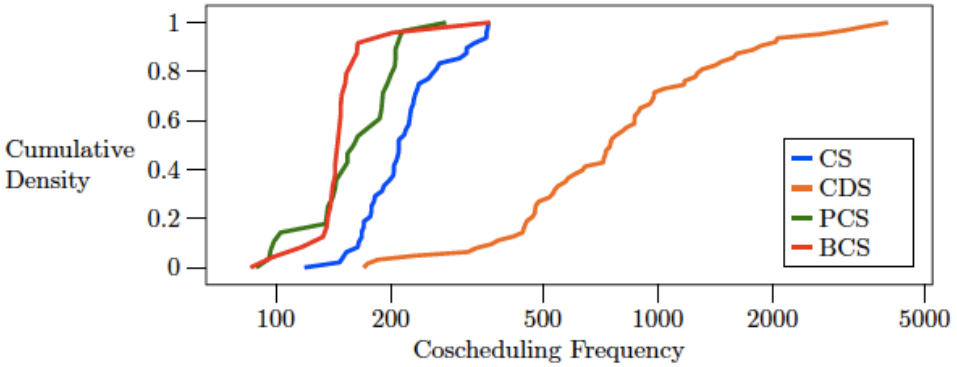
- **LU Experiment**



Time Edge Difference in BCS



(a) Solo-*CON* configuration

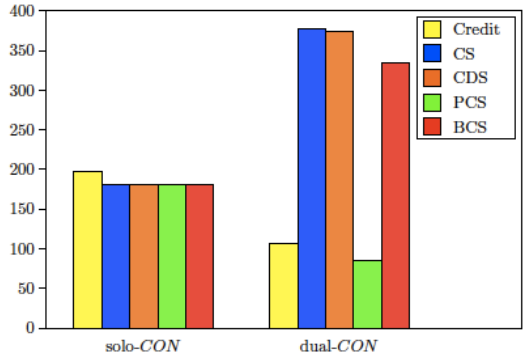


(b) Dual-*CON* configuration

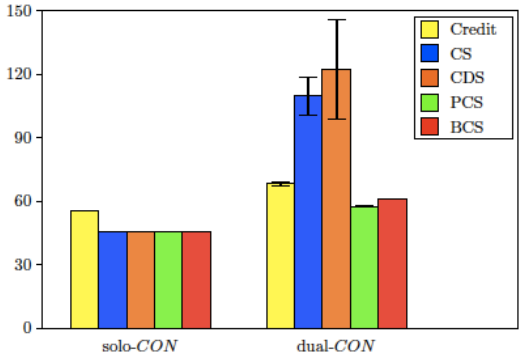
Co-scheduling Frequency

EXPERIMENTS

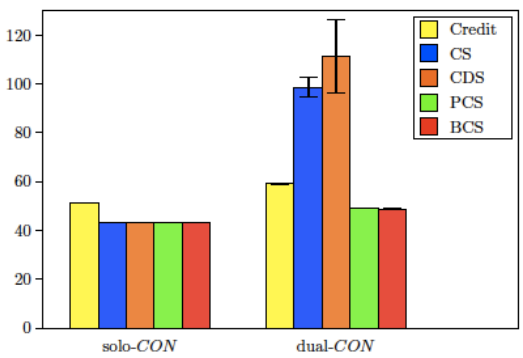
- **NPB Experiments**
  - Execution time



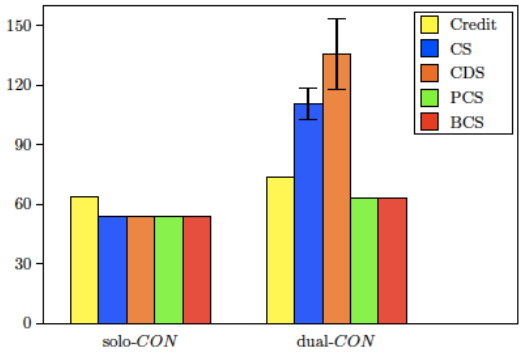
(a) BT benchmark



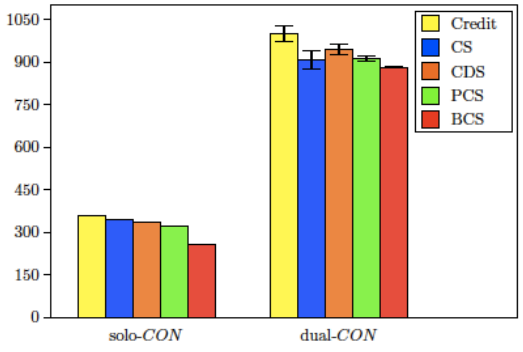
(b) CG benchmark



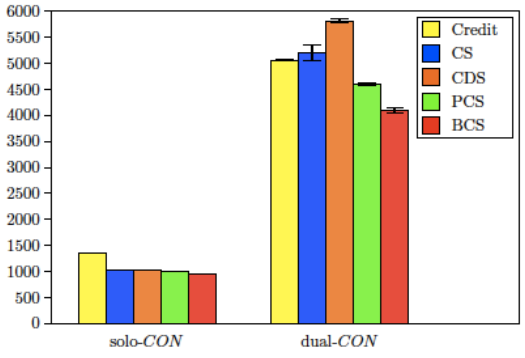
(c) EP benchmark



(d) LU benchmark



(e) FT benchmark



(f) MG benchmark

CONCLUSIONS

- **We propose two optimization schemes of hybrid co-scheduling for multiple concurrent VMs co-existing in VE**
  - PCS: Sending signals to co-scheduled VCPUs
  - BCS: Induce scheduler via priority boosting
- **Both PCS and BCS alleviate contention and exclusiveness between multiple VMs with finer space granularity**
- **Both PCS and BCS perform better in execution time and fairness than Hybrid Co-scheduling, especially when multiple concurrent VMs co-exist in system.**
- **Future Work:**
  - Remove the over-commit restriction of Co-scheduling
  - Co-scheduling in AMP Virtualized System

Hybrid Co-scheduling Optimizations for Concurrent Applications in Virtualized Environments

Yulong Yu, Yuxin Wang, He Guo and Xubin He
Dalian University of Technology & Virginia Commonwealth University

THANK YOU & ANY QUESTIONS?

REFERENCES

(Only the references in this presentation)

V. Uhlig, J. LeVasseur, E. Skoglund, et al, “Towards scalable multiprocessor virtual machines,” in Proceedings of the 3rd Virtual Machine Research and Technology Symposium, 2004, pp. 1–14.

T. Friebel and S. Biemueller. (2008) How to deal with lock holder preemption. [Online]. Available: http://www.amd64.org/fileadmin/user upload/pub/2008-Friebel-LHP-GI OS.pdf

W. Jiang, Y. Zhou, Y. Cui, et al, “CFS optimizations to KVM threads on multi-core environment,” in International Conference on Parallel and Distributed Systems, 2009, pp. 348–354.

C. Weng, Z. Wang, M. Li, et al, “The hybrid scheduling framework for virtual machine system,” in Virtual Execution Environments, 2009, pp. 111–120.

D.G. Feitelson and L. Rudolph, “Gang scheduling performance benefits for fine-grain synchronization,” Journal of Parallel and Distributed Computing, vol. 16, no. 1, pp. 306–318, 1992.

VMWare Communities. (2008) Co-scheduling smp vms in vmware esx server. [Online]. Available: http://communities.vmware.com/docs/DOC-4960

C. Xu, Y. Bai and C. Luo, “Performance evaluation of parallel programming in virtual machine environment,” in International Conference on Network and Parallel Computing, 2009, pp. 140–147.

Y. Bai, C. Xu and Z. Li, “Task-aware based co-scheduling for virtual machine system,” in Symposium On Applied Computing, 2010, pp. 181–188.

V. Kazempour, A. Kamali and A. Fedorova, “AASH: an asymmetry-aware scheduler for hypervisor,” in Virtual Execution Environments, 2010, pp. 85–96.

ACKNOWLEDGEMENT

This work is partially supported by the Sea Sky Scholar fund of the Dalian University of Technology.

The author He's research is sponsored in part by National Science Foundation grant CCF-1102624.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies.