

# BLUE WATERS

SUSTAINED PETASCALE COMPUTING

## Characterizing the Influence of System Noise on Large-Scale Applications by Simulation

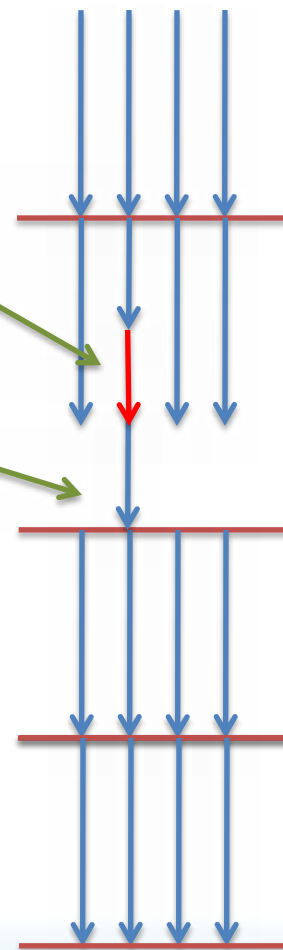
Torsten Hoefler, Timo Schneider, Andrew Lumsdaine



GREAT LAKES CONSORTIUM  
FOR PETASCALE COMPUTATION

# System Noise – Introduction and History

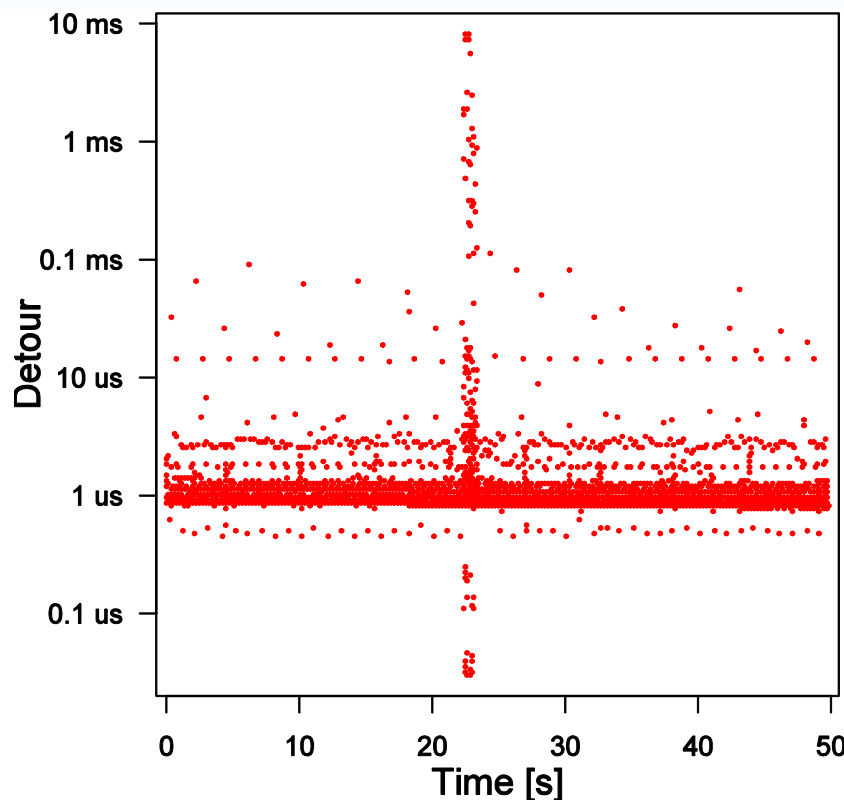
- CPUs are time-shared
  - Deamons, interrupts, etc. steal cycles
  - No problem for single-core performance
    - Maximum seen: 0.26%, average: 0.05% overhead
  - “Resonance” at large scale (Petrini et al '03)
- Numerous studies
  - Theoretical (Agarwal'05, Tsafrir'05, Seelam'10)
  - Injection (Beckman'06, Ferreira'08)
  - Simulation (Sottile'04)



## Measuring OS Noise on a Single Core

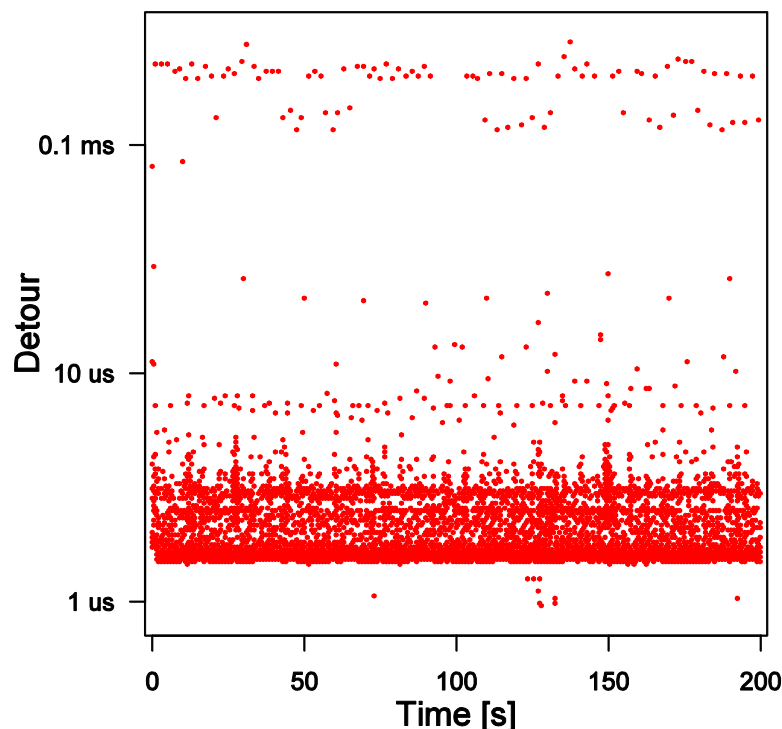
- Selfish Detour Benchmark (Beckman et al.)
  - Tight execution loop, benchmark iteration time
  - Record each outlier in iteration time
  - Improved detour (~30% better resolution)
- Detour implemented in Netgauge benchmark tool
  - Also FWQ, FTQ (not used in this work)
  - Available at: <http://www.unixer.de/Netgauge>

## Measurement Results – CHiC Linux (diskless)



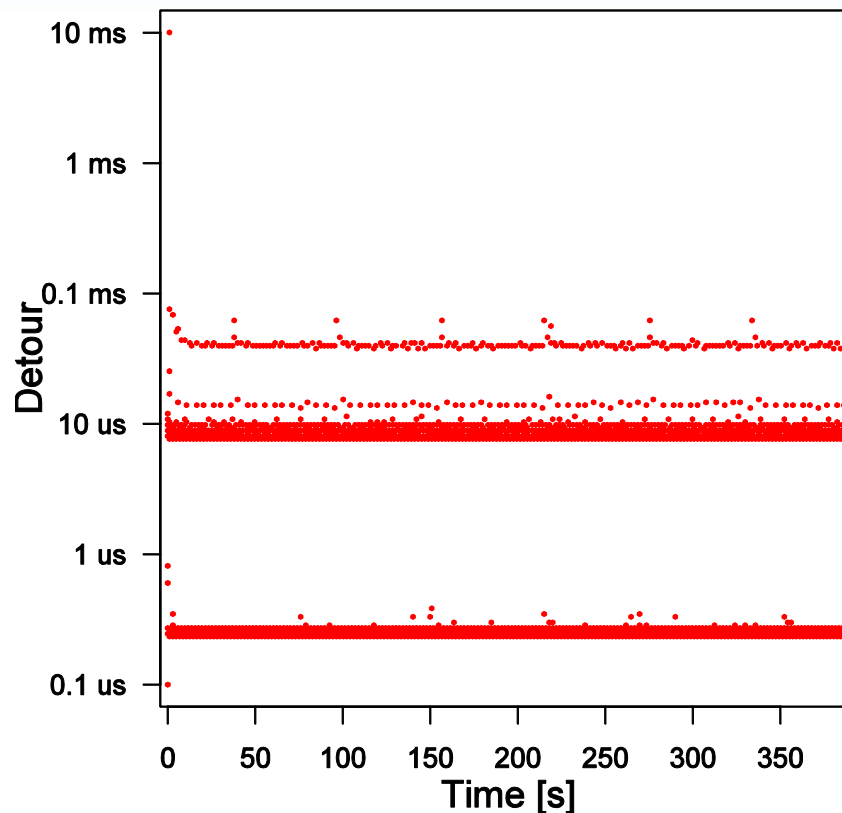
- 2152 Opteron cores, 11.2 Tflop/s Linux 2.6.18
- Resolution: 3.74 ns, noise overhead: 0.21%

## Measurement Results – SGI Altix



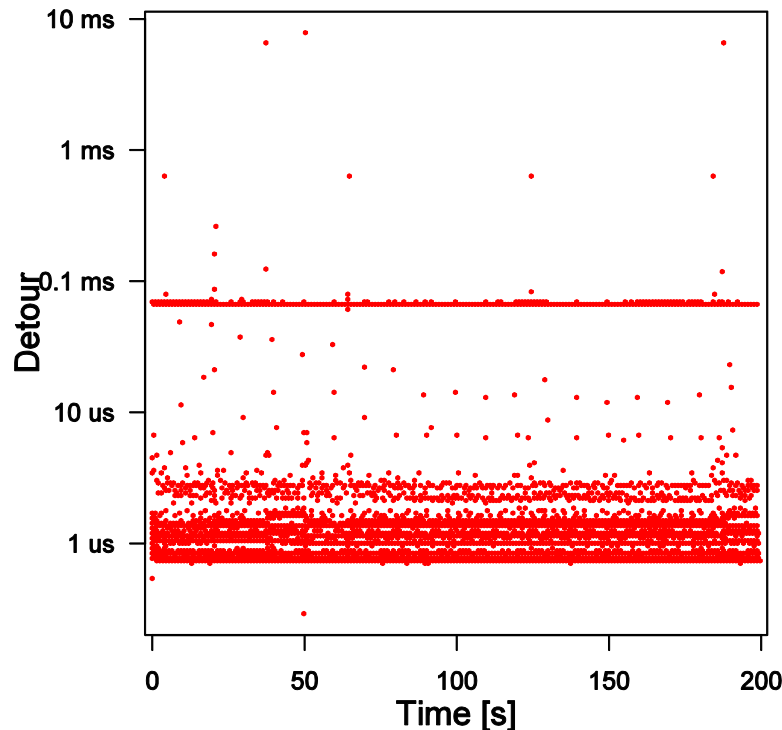
- Altix 4700, 2048 Itanium II cores, 13.1 Tflop/s, Linux 2.6.16
- Resolution: 25.1 ns, noise overhead: 0.05%

## Measurement Results – BG/P ZeptoOS



- 164k PPC 450 cores, 485.6 Tflop/s, ZeptoOS 2.6.19.2
- Resolution: 29.1 ns, noise overhead: 0.08%

## Measurement Results – Cray XT-4 (Jaguar)



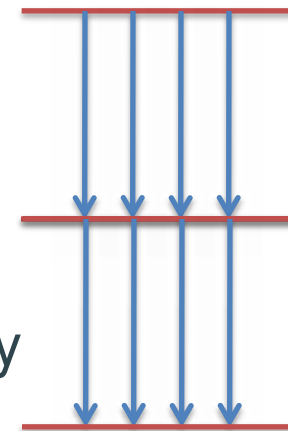
- 150k Opteron cores, 1.38 Pflop/s, Linux 2.6.16 CNL
- Resolution: 32.9 ns, noise overhead: 0.02%

## An Analytical Model for Noise Propagation

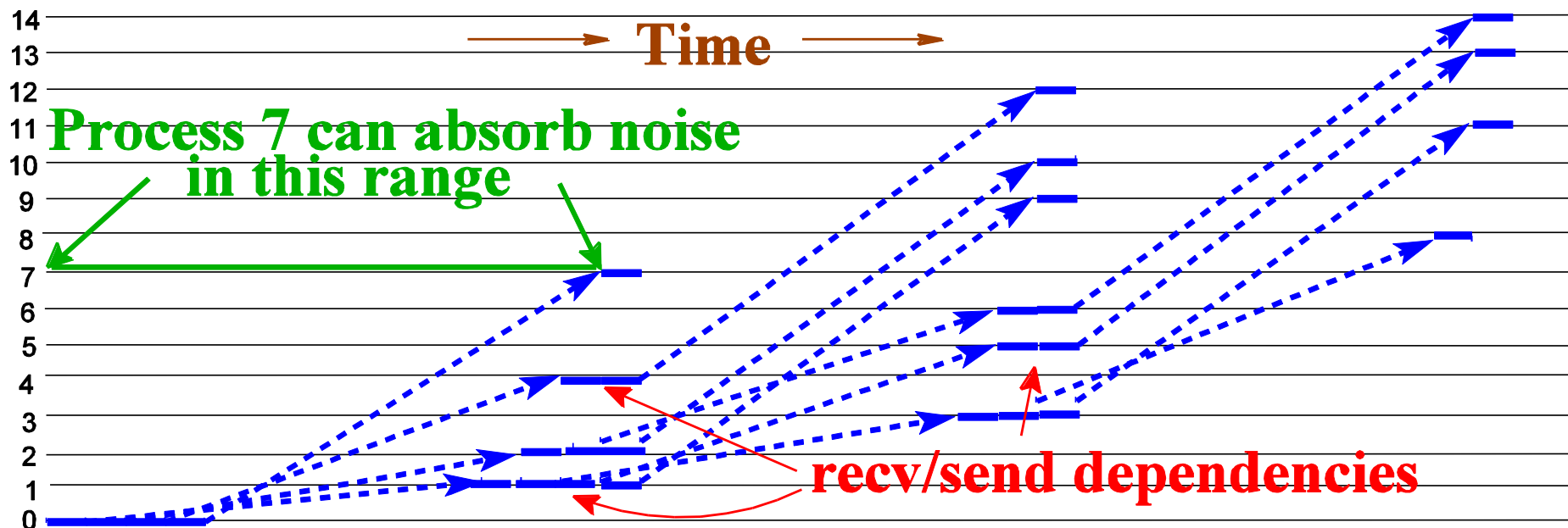
- Synchronization propagates or absorbs noise
  - Lamport's happens-before-relation for messages
  - Depends on relative time of send/recv (or wait)
- Several protocol-specific details
  - Small (eager), large (rendezvous), and nonblocking
- LogP model to express communication
  - Several missing pieces
  - LogGPS model (Ino et al.) captures most effects!
  - We added "O" to capture s/r overhead per byte

# Collective Operations

- MPI-2.2: “[...] a collective communication call may, or may not, have the effect of synchronizing all calling processes. This statement excludes, of course, the barrier function.”
- Main weaknesses in theoretical models:
  - Assumption 1: All collective operations synchronize
    - In fact, many do not (e.g., Bcast, Scan, Reduce, ...)
  - Assumption 2: Collectives synchronize instantaneously
    - In fact, they (most likely) communicate with messages
  - Assumption 3: All processes leave collective simultaneously
    - In fact, they leave as early as possible (when data is consistent)

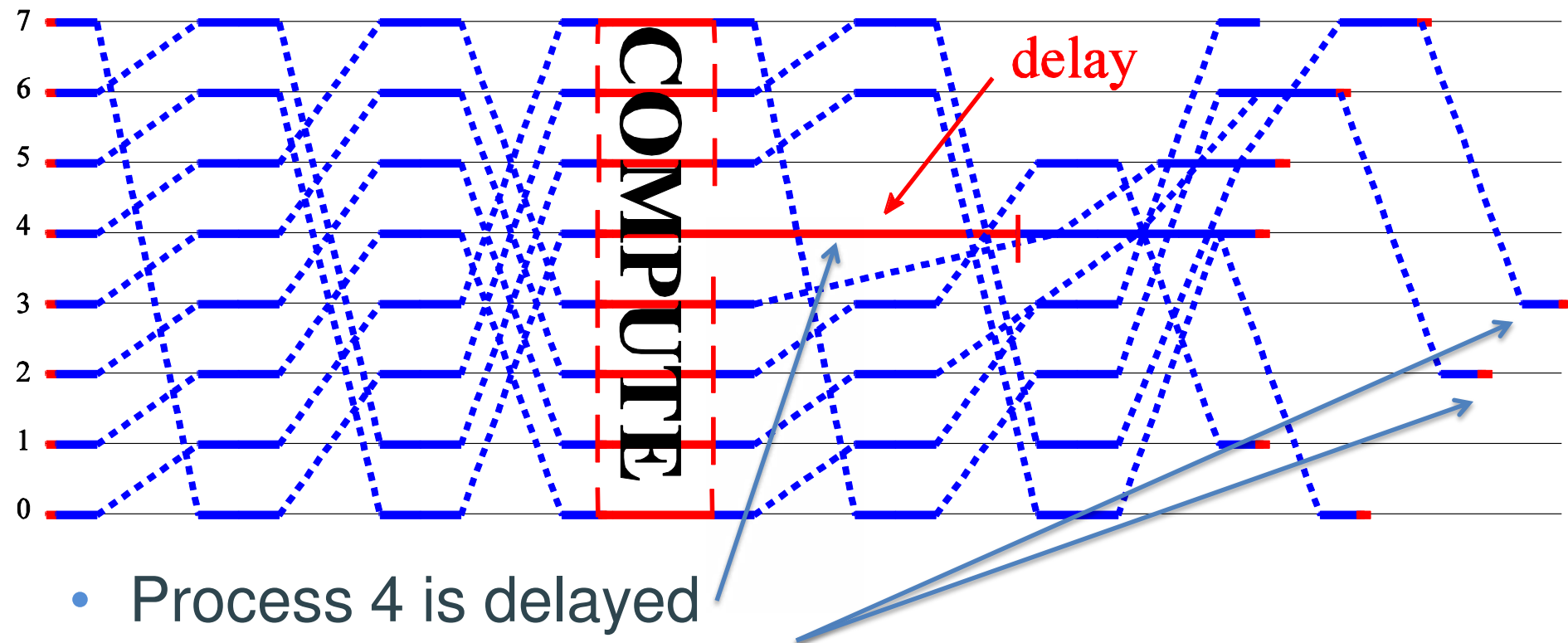


## Example: Binomial Broadcast Tree



- Violates all three assumptions:
  - No global or instant synchronization, asynchronous exit

## A Noisy Example – Dissemination Barrier

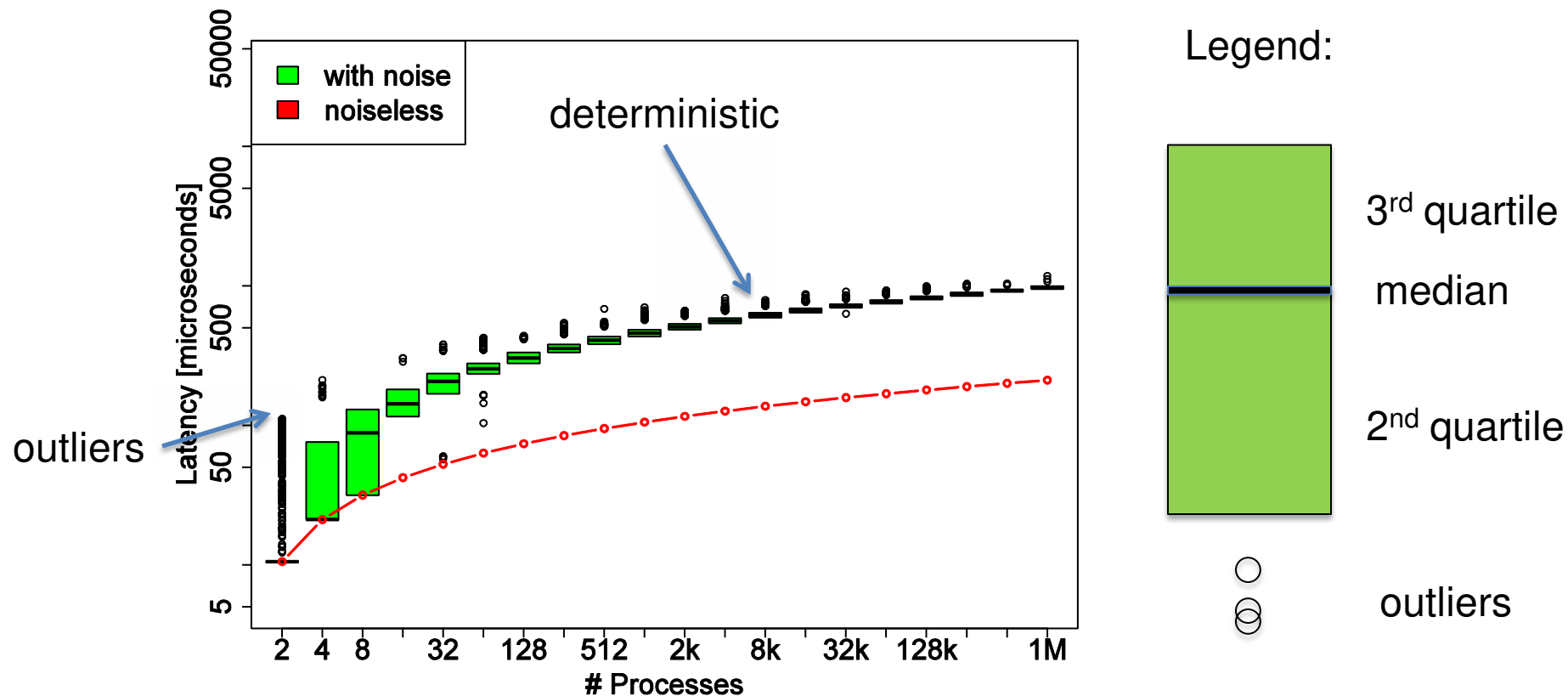


- Process 4 is delayed
  - Noise propagates “*wildly*” (of course deterministic)

# LogGOPS Simulation Framework

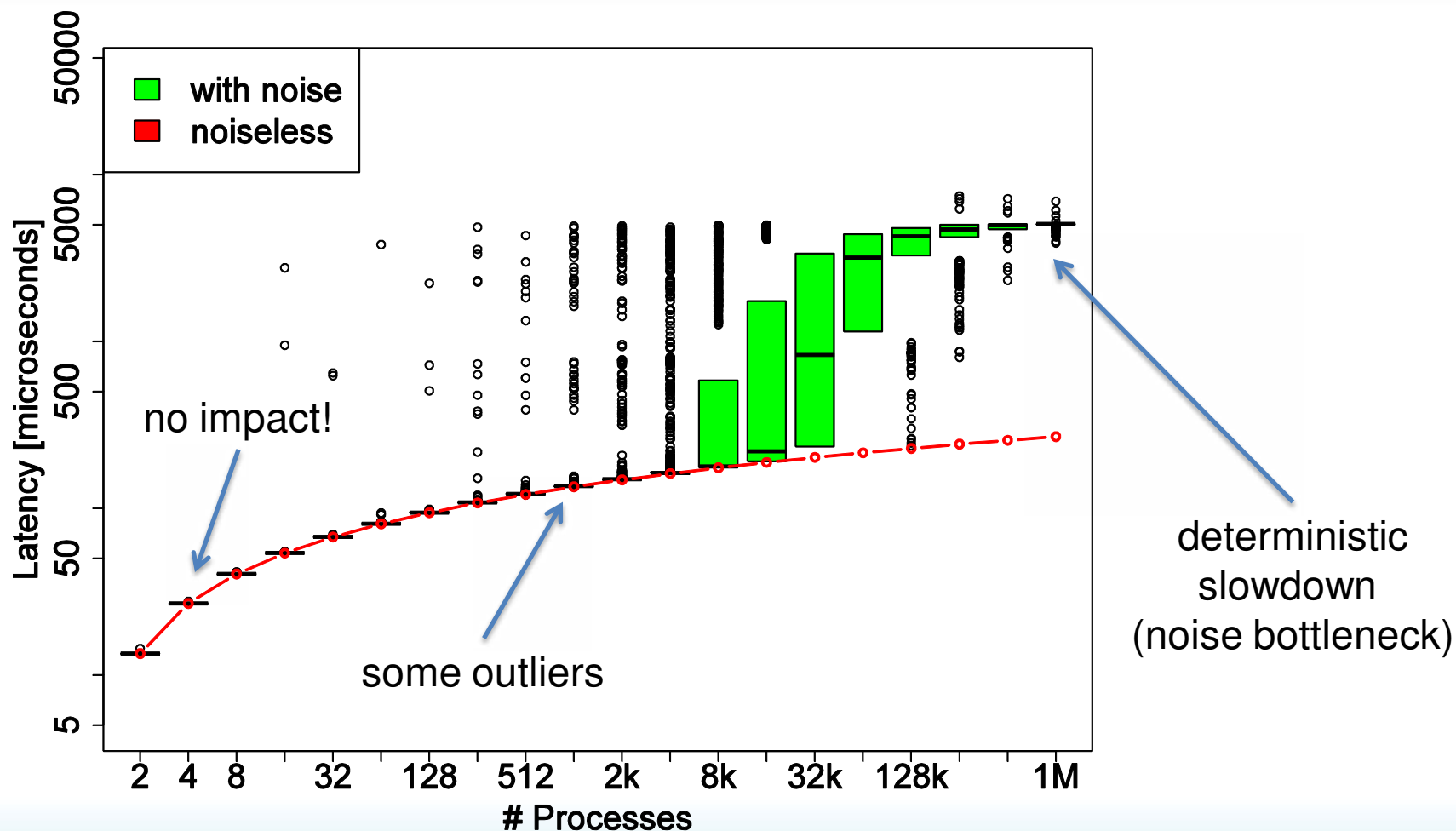
- Detailed analytical modeling is hard!
- Model-based (LogGOPS) simulator
  - Available at: <http://www.unixer.de/LogGOPSim>
  - Discrete-event simulation of MPI traces (<2% error) or collective operations (<1% error)
  - >  $10^6$  events per second!
- Allows for trace-based noise injection
  - In  $o_s$ ,  $o_r$ ,  $O$ , local reduction, and application time
- Validation
  - Simulations reproduce measurements by Beckman and Ferreira well!
- Details: Hoefler et al. LogGOPSim – Simulating Large-Scale Applications in the LogGOPS Model (Workshop on Large-Scale System and Application Performance, Best Paper)

# Single Collective Operations and Noise

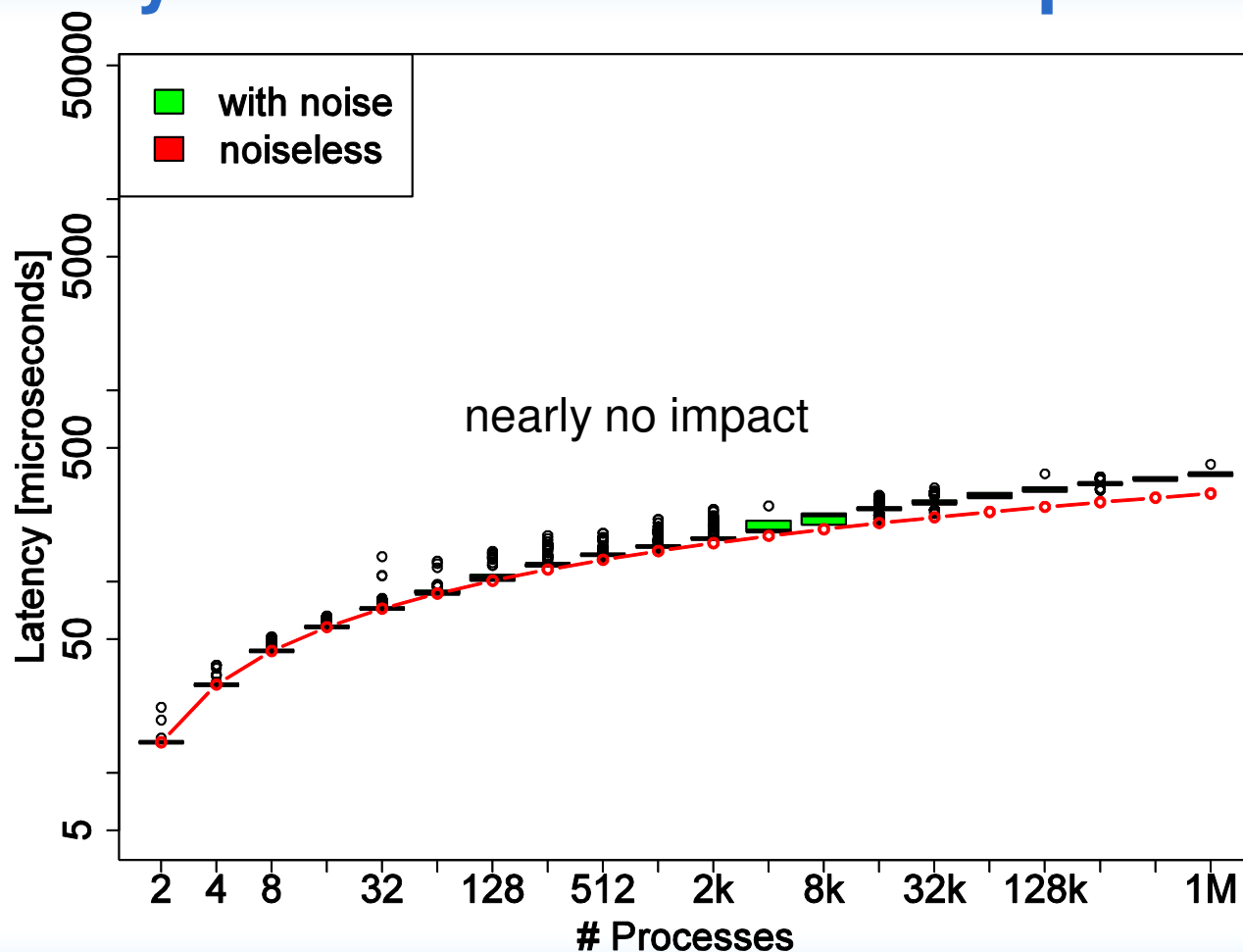


- 1 Byte, Dissemination, regular noise, 1000 Hz, 100  $\mu$ s

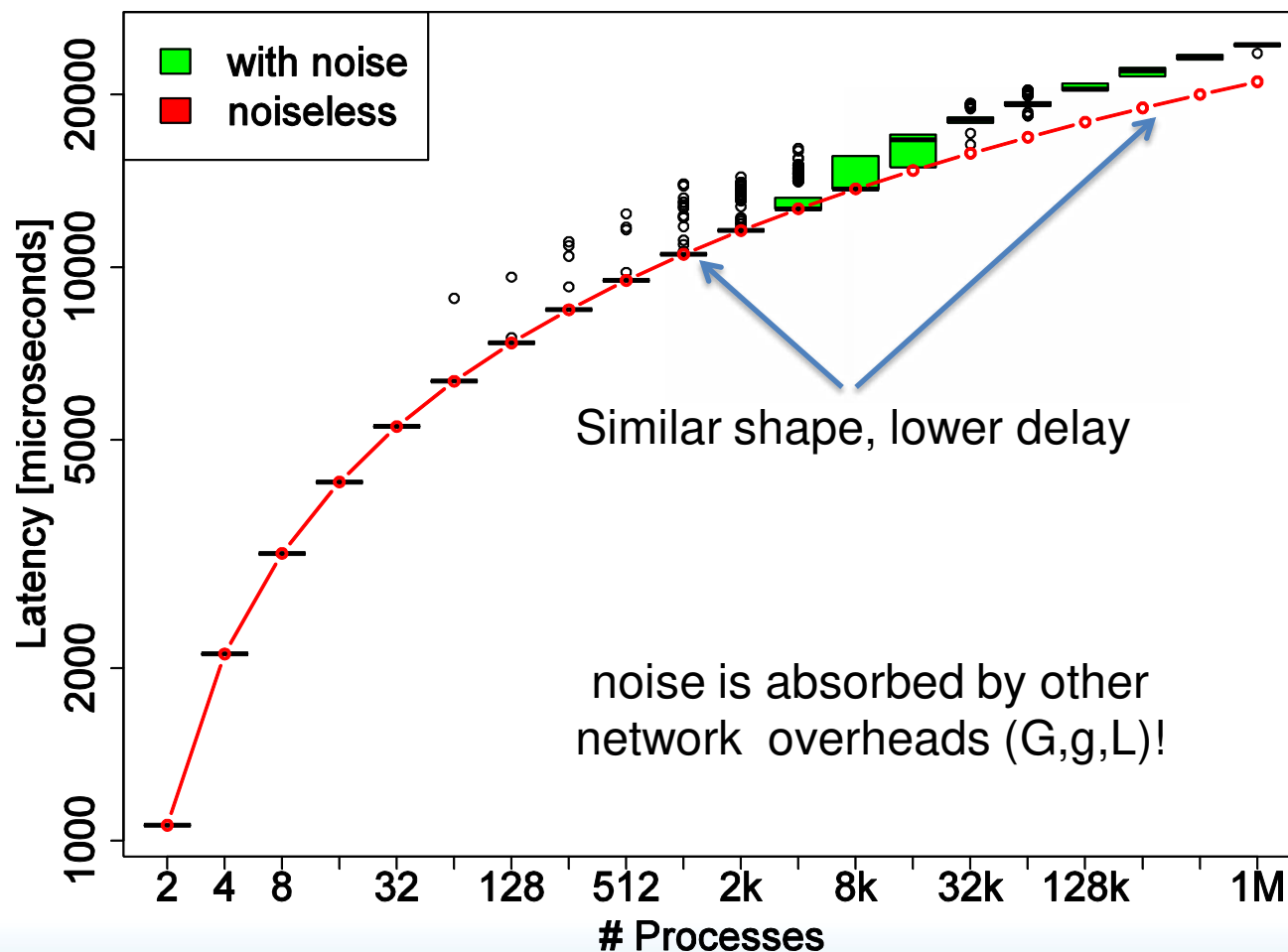
# Single Byte Dissemination on Jaguar



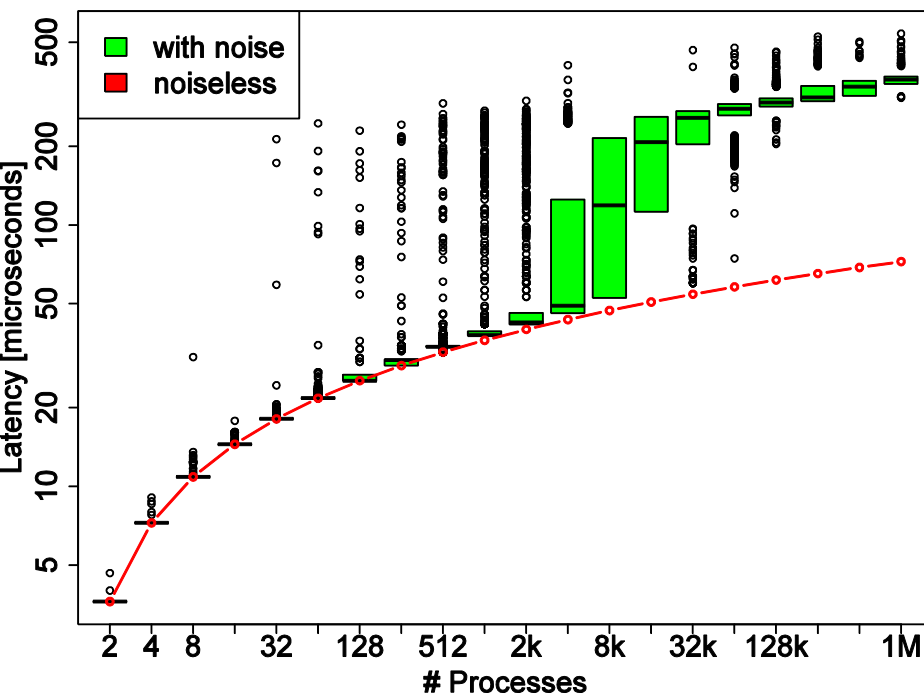
# Single Byte Dissemination on ZeptoOS



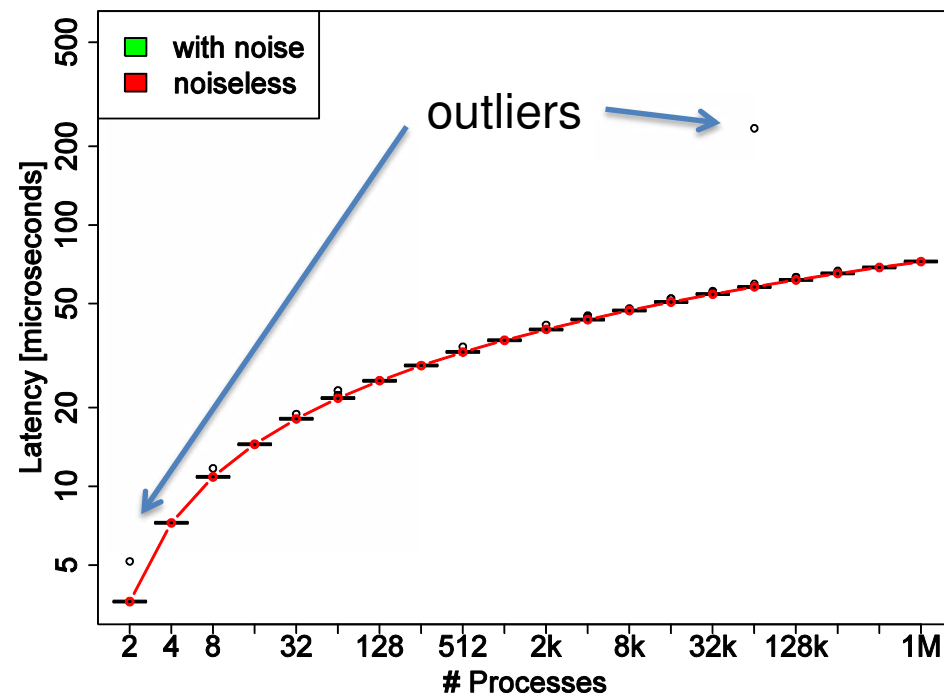
# 1MiB Messages on Jaguar



## Effect of Co-Scheduling Noise (Altix)

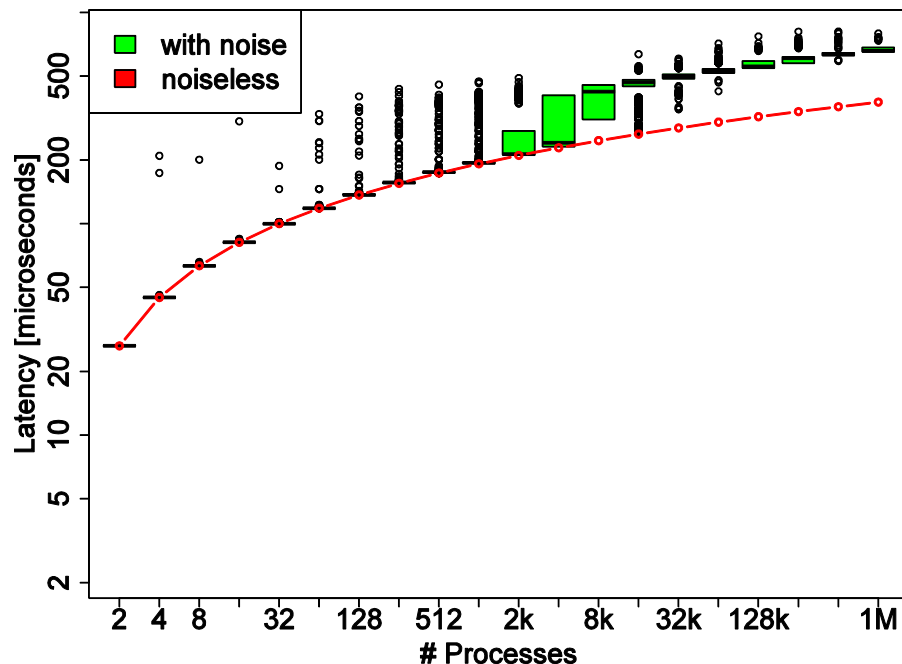


Normal



Co-Scheduled

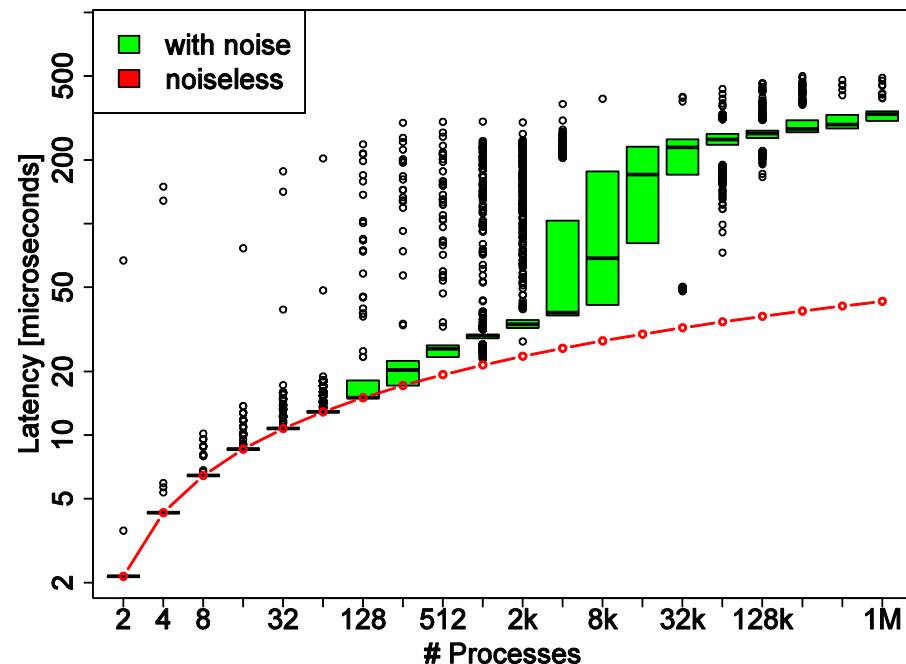
# Does the Network Speed Matter?



0.1x network speed

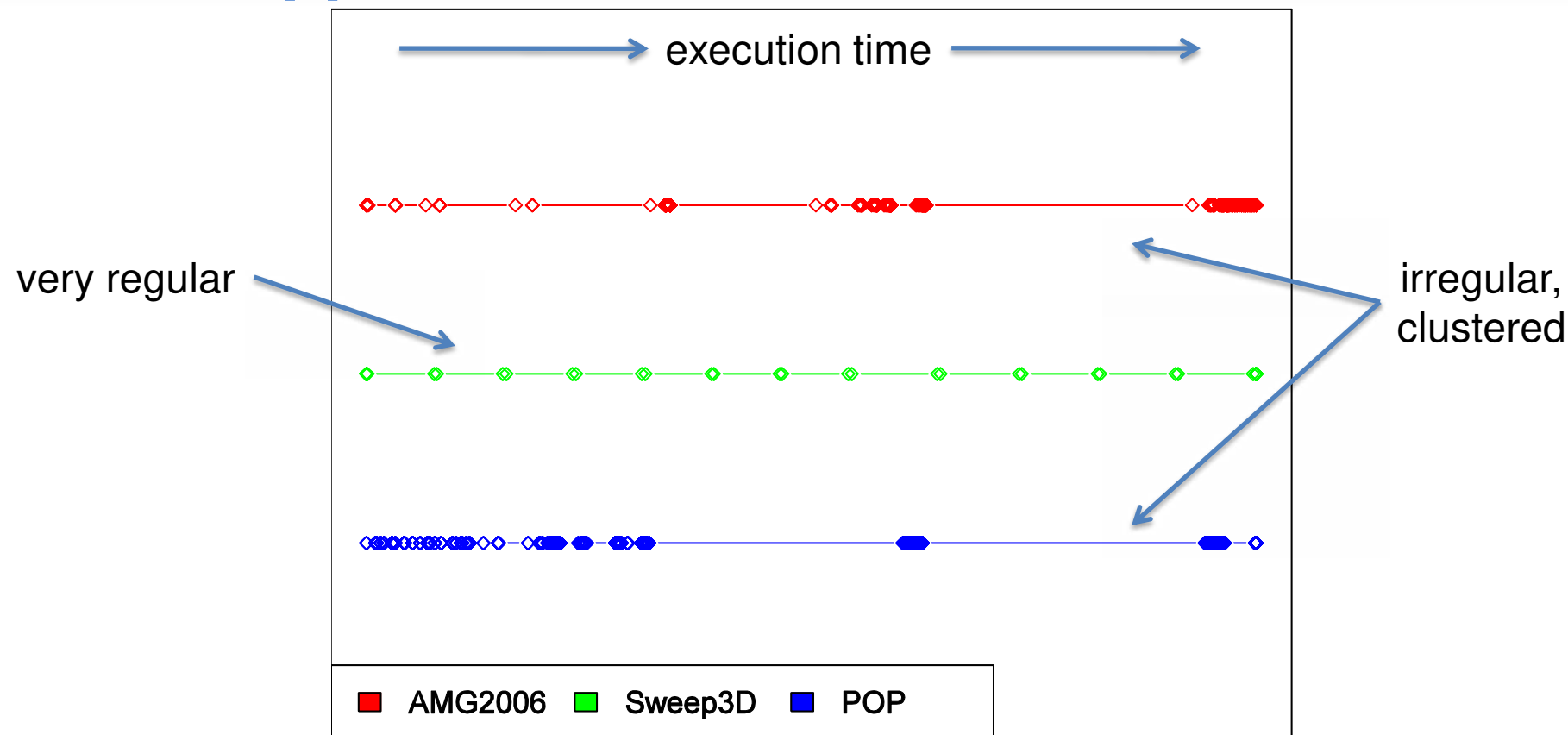
Method: increase/decrease L,G,g

Observation: noise bottleneck independent of network speed



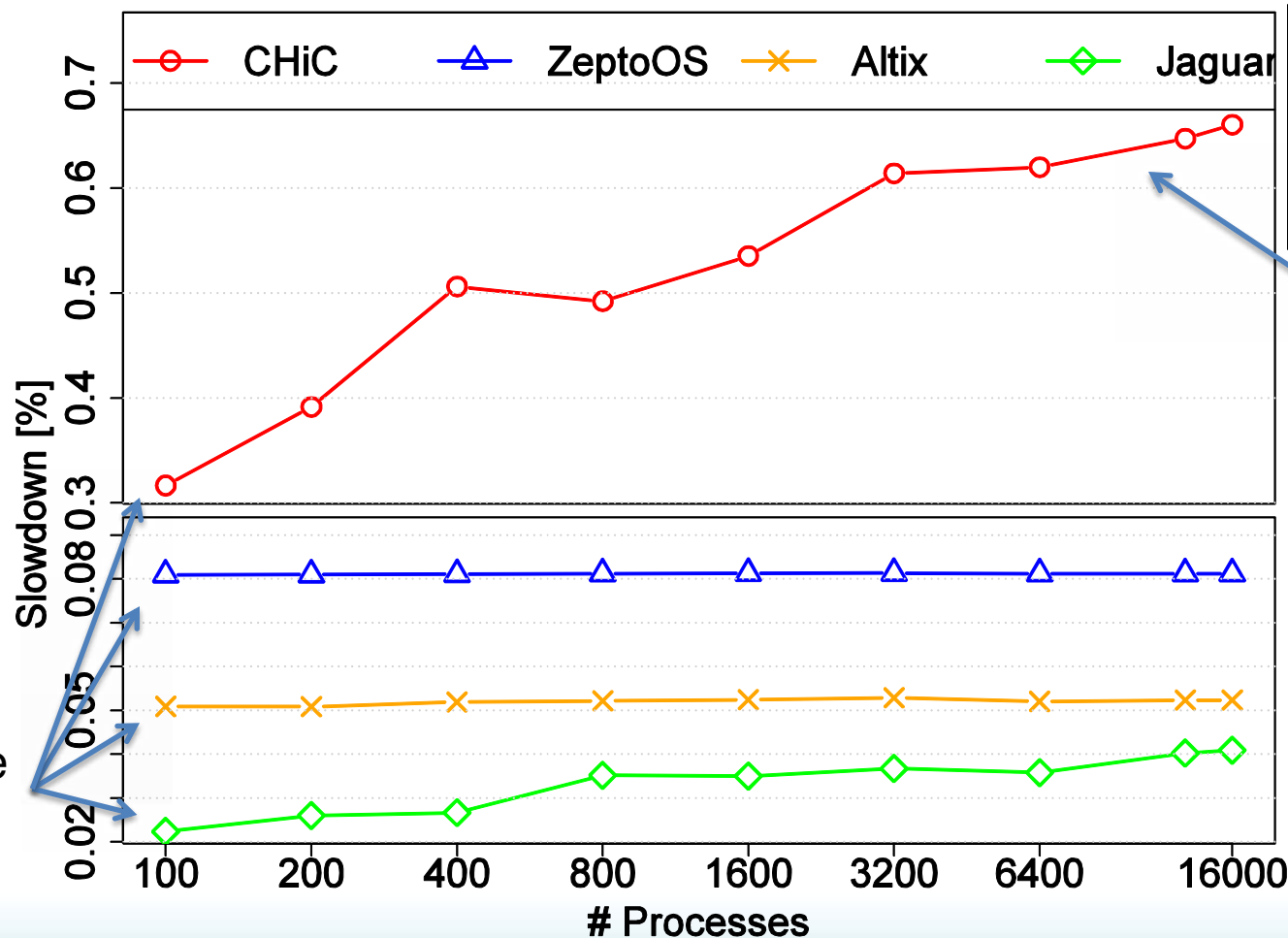
10x network speed

# Real Applications



## Distribution of Collective Operations

# Sweep3D (Collective and Point-to-Point)



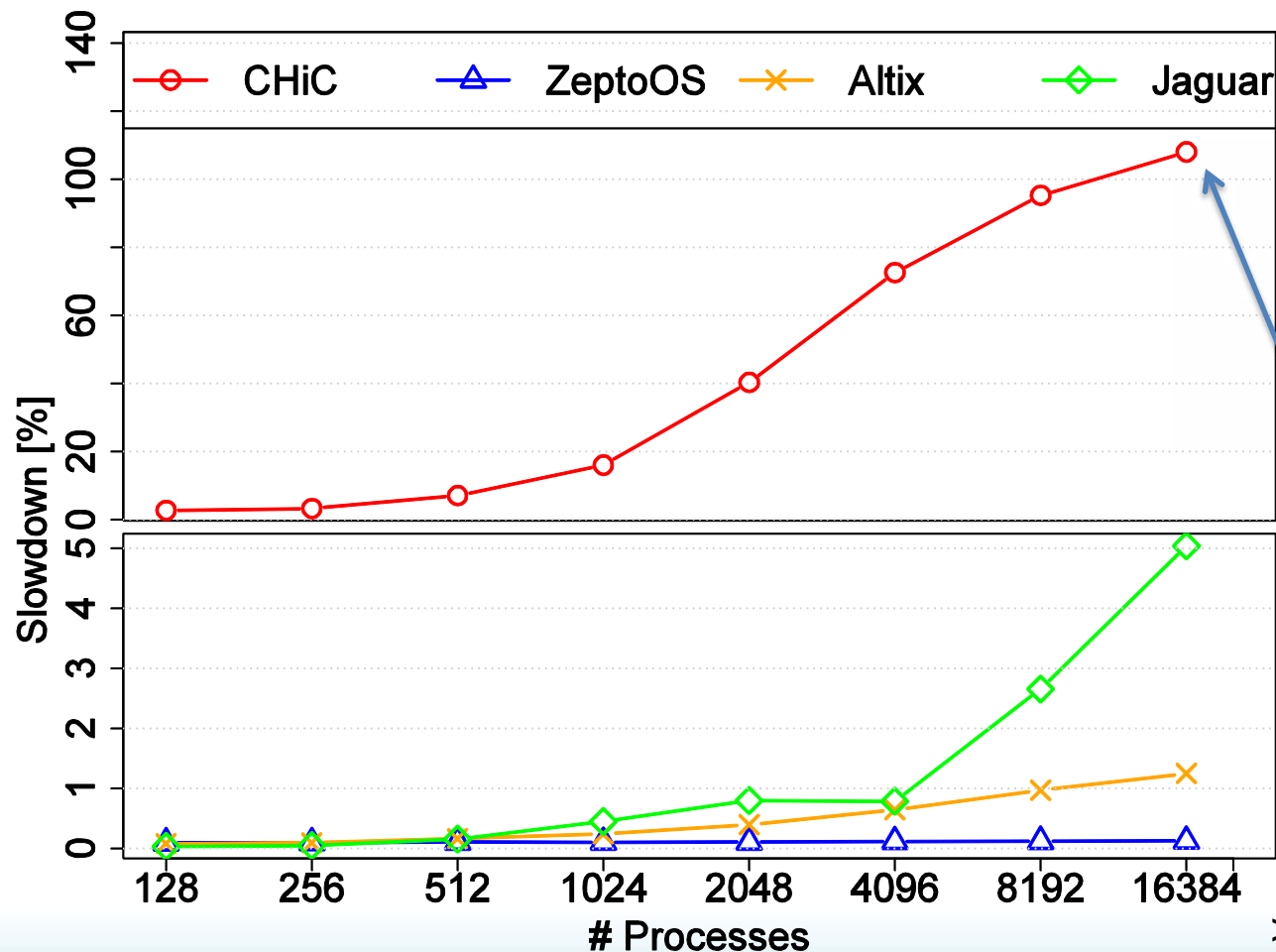
92.4% bl p2p  
4 bcast  
3 barrier  
32 allreduce

<0.7%

serial noise  
overheads

>289 million  
messages!

# POP (Collective and Point-to-Point)



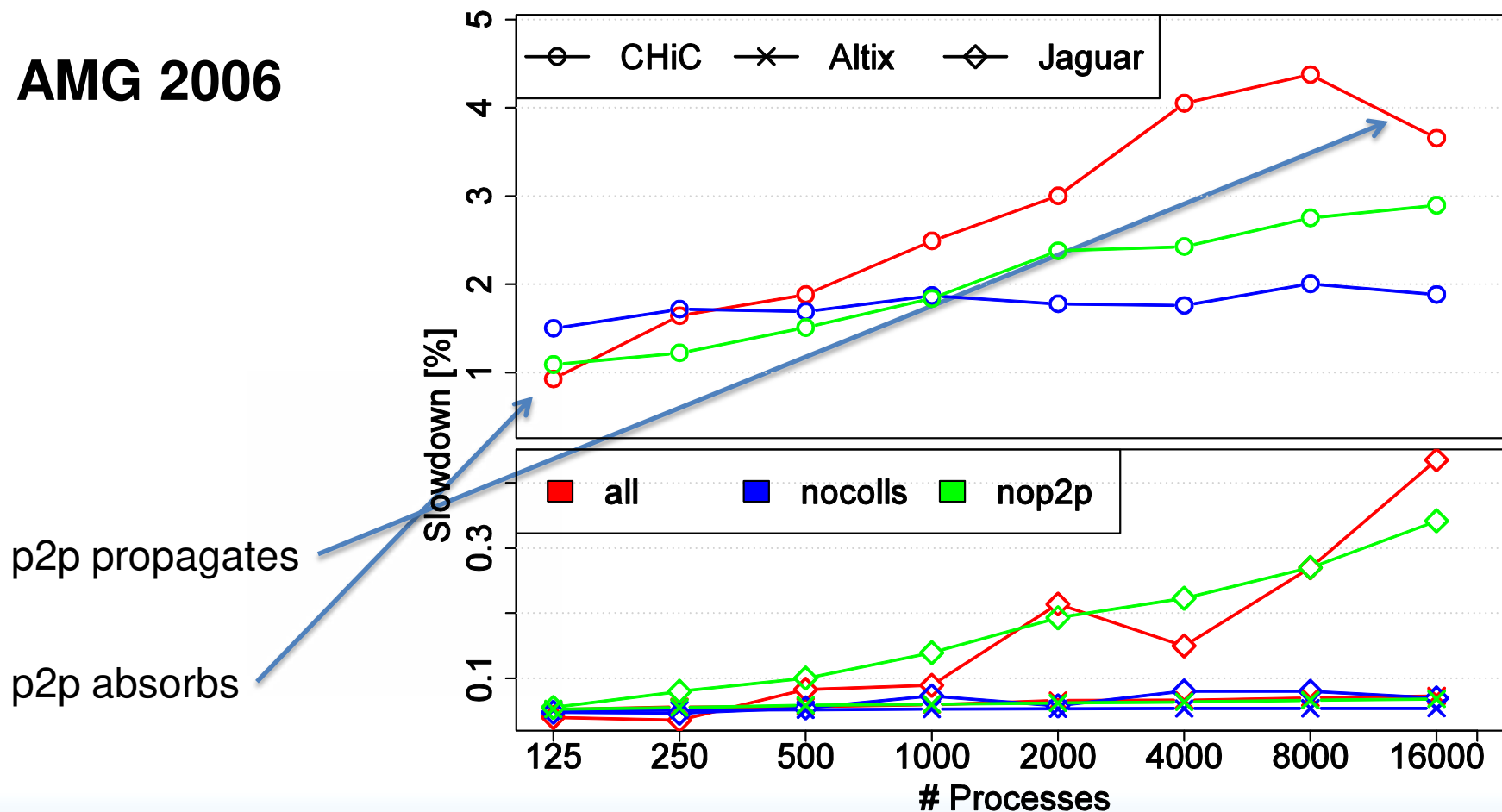
0.2% nb p2p  
703 bcast  
575 barrier  
608 allreduce

>2x slowdown!

>625 million messages!

# Does Point-to-Point Communication Matter?

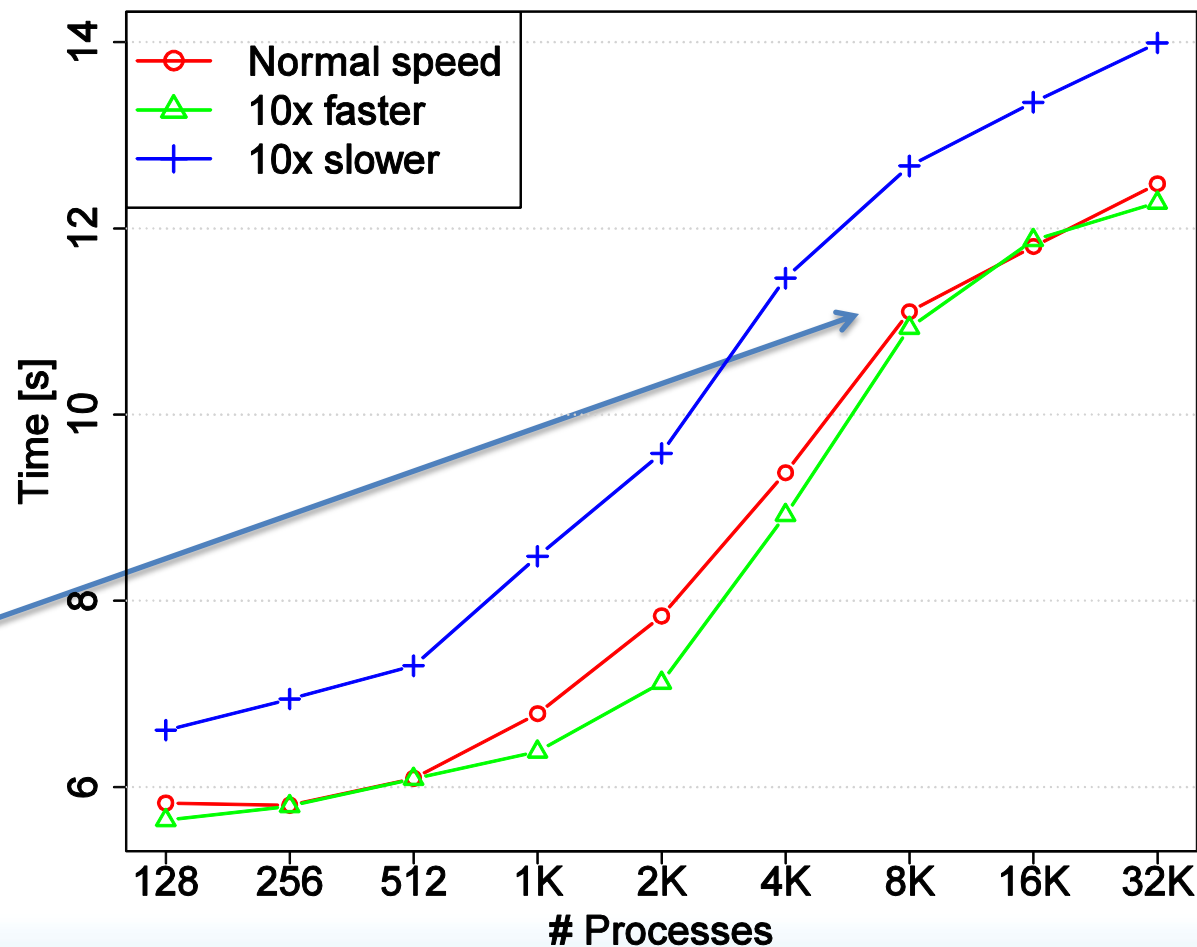
## AMG 2006



# Influence of Network Speed on Applications

POP @ CHiC


Noise bottleneck:  
faster network  
does not increase  
performance



## Conclusions & Future Work

- Modeling OS noise is not that simple
  - Will validate used models with simulation
- Model-based simulation approach scales well
  - Results match previous benchmark studies (<6% error)
- Overhead depends on noise *shape* rather than *intensity*
  - ZeptoOS shows nearly no propagation! (0.08% overhead)
  - Cray XT is severely impacted! (0.02% overhead)
- Noise bottleneck is serious at scale!
  - Faster network or CPU cannot help, noise will dominate!
- We developed a tool-chain to adjust the bottleneck
  - Available online: <http://www.unixer.de/LogGOPSim>

# Collaborators, Acknowledgments & Support

- Co-Authors:
  - Timo Schneider, Andrew Lumsdaine  **INDIANA UNIVERSITY**  
PERVASIVE TECHNOLOGY INSTITUTE
- Thanks to (alphabetically)
  - Franck Cappello, Steven Gottlieb, William Gropp, William Kramer, and Marc Snir
- Sponsored by

