Sino-German Workshop on Multiphysics Device Simulation and Hardware-Aware Computing

Xi'An, China, October 11, 2024

Friedrich-Alexander-Universität Erlangen-Nürnberg

Analytical performance modeling for HPC workloads

Georg Hager

Erlangen National High Performance Computing Center (NHR@FAU)

Analytical, Resource-Based, First-Principles Performance Models

A mathematical representation of hardware-software interaction based on simplified machine and application models, which predicts the performance or runtime of a program using hardware resource limits and code requirements

$$S(N) = \frac{1}{s + \frac{1-s}{N} + c(N)}$$

Amdahl's Law with communication

$$T_{PtP} = T_l + \frac{L}{B}$$

Hockney model for message transmission time

$$T_{\text{exec}} = f(T_{nOL}, T_{\text{data}}, T_{OL})$$

ECM model for loop code execution time

$$T_{\text{exec}} = \max(T_{\text{calc}}, T_{\text{data}})$$

Roofline model for loop code execution time

Motivation

Analytic performance model:

A mathematical representation of hardware-software interaction based on simplified machine and application models, which predicts the performance or runtime of a program using hardware resource limits and code requirements

- Basic questions addressed by analytic performance models
  - What is the bottleneck?
  - What is the next bottleneck after optimization?
  - Impact of hardware features → co-design, architectural exploration
- What if the model fails?
  - We learn something
  - We may still be able to use the model in a less predictive way

Performance Engineering (PE)

HPC workloads and “bottleneckology”

A general view on resource bottlenecks

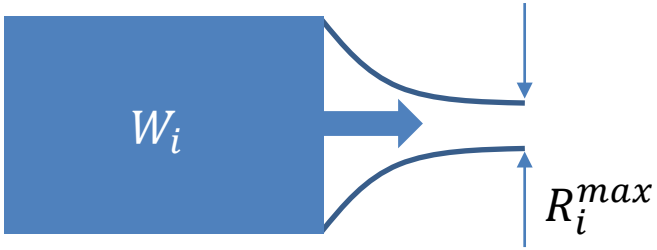
Maximum performance of a steady-state code when limited by a bottleneck?

- Resource bottleneck i delivers resources at maximum rate R_i^{max}
- W_i = needed amount of resources

Machine property

Application property

- Minimum runtime: $T_i = \frac{W_i}{R_i^{max}} + \lambda_i$



Multiple bottlenecks

- Multiple bottlenecks → multiple min. runtimes:

$$T_{\text{expect}} = f(T_1, \dots T_n)$$

- Overall performance: $$P_{\text{expect}} = \frac{W}{T_{\text{expect}}}$$

Bottleneck models for execution time

How do we reconcile the multiple bottlenecks?
I.e., what is the functional form of $f(T_1, \dots T_n)$?

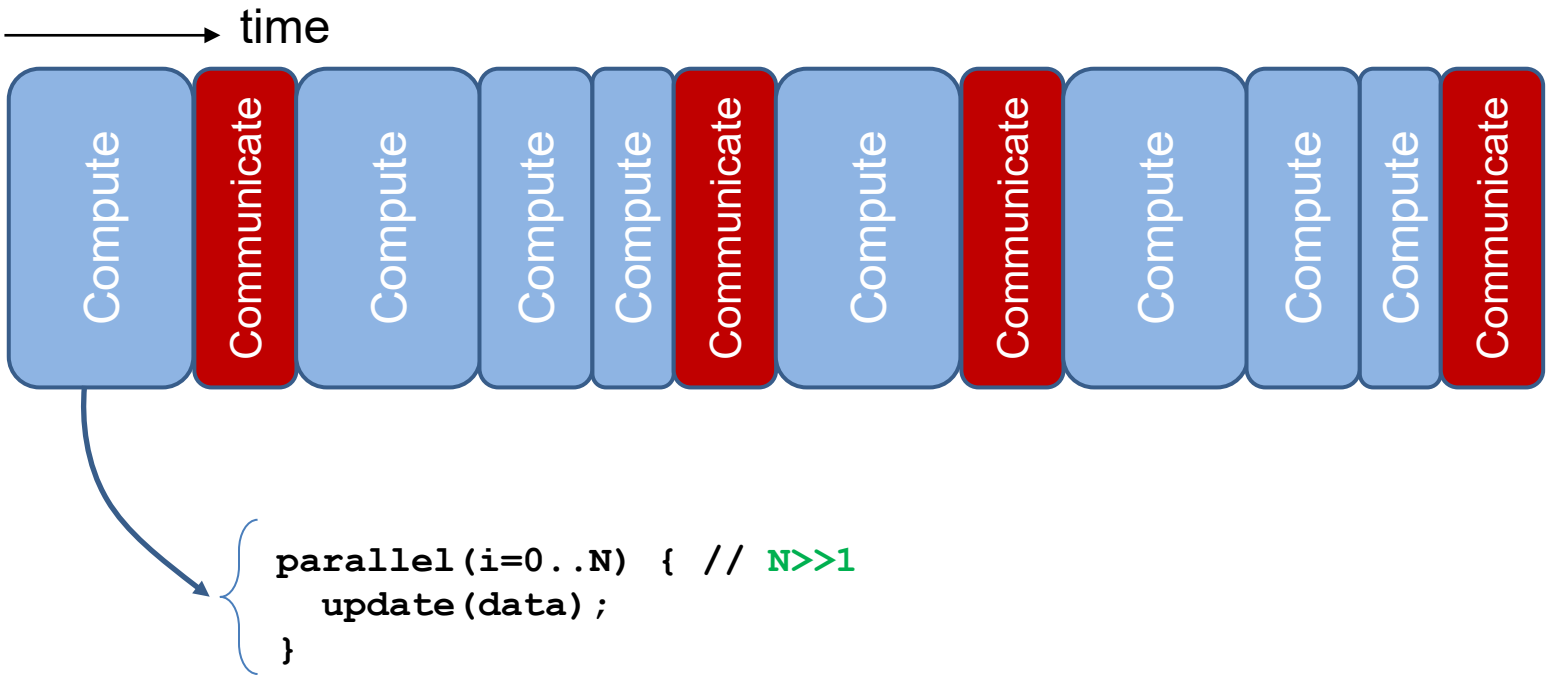
→ pessimistic model (no overlap): $f(T_1, \dots T_n) = \sum_i T_i$
→ optimistic model (full overlap): $f(T_1, \dots T_n) = \max(T_1, \dots T_n)$

Roofline model
(Hockney et al., 1989
Williams et al., 2008)

Naïve Roofline model (simplest case):

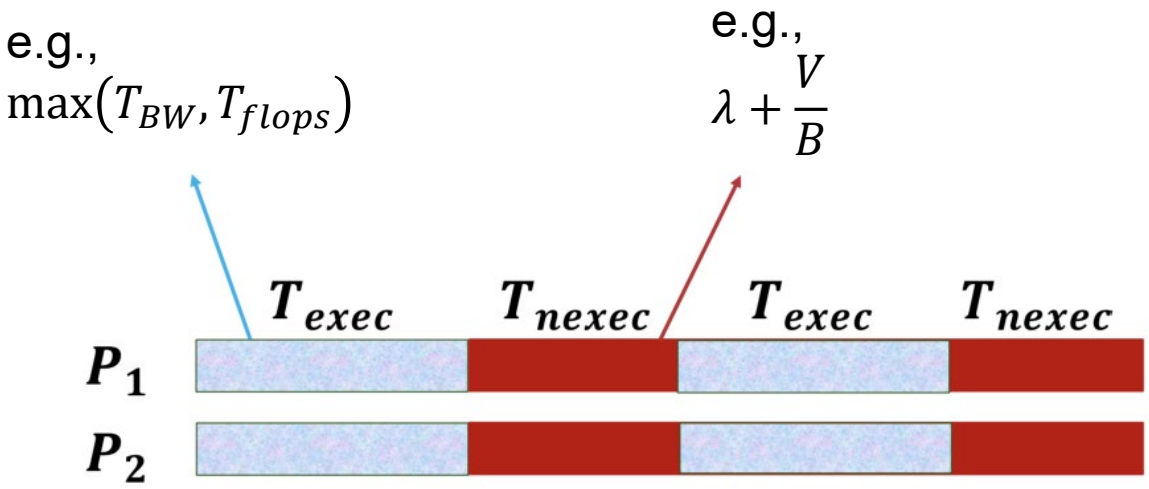
$$P_{pred} = \min\left(R_{flops}, R_{BW} \times \frac{W_{flops}}{W_{BW}}\right)$$

General application pattern: multiple phases



Composite analytic models: simple case

Plausible assumption: $T = T_{exec} + T_{nexec}$



In practice, $T \neq T_{exec} + T_{nexec}$ and it can go in either direction

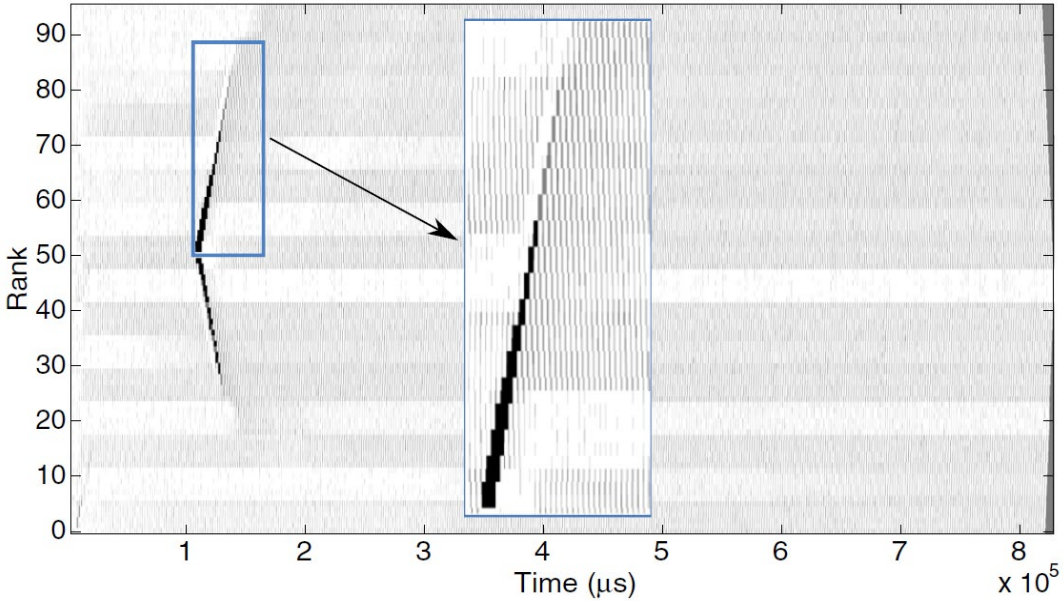
Idle wave progression

Markidis et al. (2015)

Simulator-based analysis

Idle waves perceived as "damped linear waves"

Classical wave equation postulated for continuum description

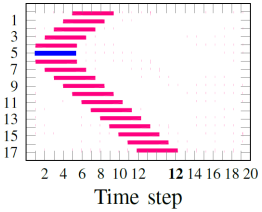


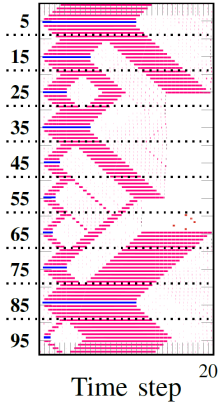
S. Markidis et al.: *Idle waves in high-performance computing*. Phys. Rev. E **91**(1), 013306 (2015).
DOI: 10.1103/PhysRevE.91.013306

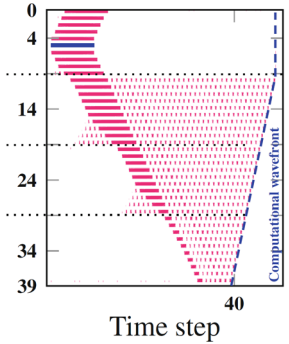
Research questions

Setting: MPI- or hybrid-parallel bulk-synchronous barrier-free programs

- How do “disturbances” propagate?
  - Injected idle periods
  - Dependence on communication characteristics
- How do idle waves interact with each other, with noise, and with the hardware?
  - Idle wave decay
    (noise-induced, bottleneck-induced, topology-induced)
- How do computational waves form? Instabilities?
  - Core-bound vs. memory-bound
  - Amplitude of the computational wave?
- Continuum description?







Idle wave propagation speed for scalable code

Assumptions:

- Scalable code (on ccNUMA domain)
- alternates between execution and communication phases
- has inter-process dependencies via point-to-point communication

Simplest case: Next-neighbor (e.g., 1-D halo) communication

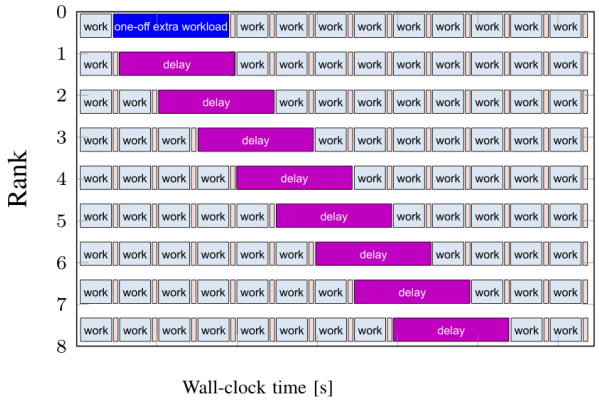
$$v_{\text{silent}}^{\text{min}} = 1 \left[\frac{\text{ranks}}{\text{iter}}\right] \times \frac{1}{T_{\text{exec}} + T_{\text{comm}}} \left[\frac{\text{iter}}{\text{s}}\right]$$

Number of communication partners and details of communication grouping influence the speed

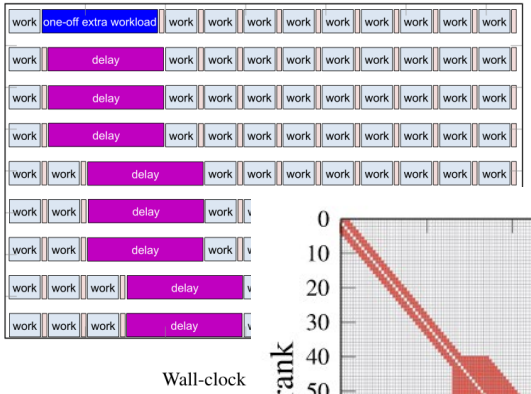
DOI: 10.1007/978-3-030-78713-4_19

Communication topology and idle wave speed

Long-distance point-to-point communication → fast idle waves

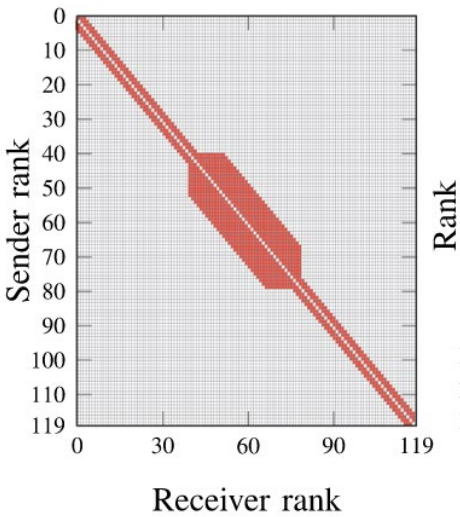


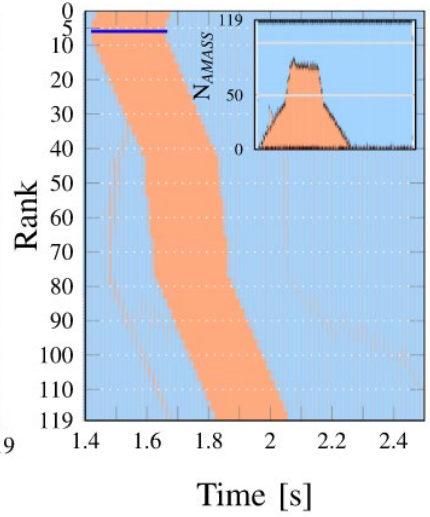
(a) $d = \pm(1)$ (slow idle wave)



(b) $d = \pm(1, 2)$ (thr

arXiv:2205.04190

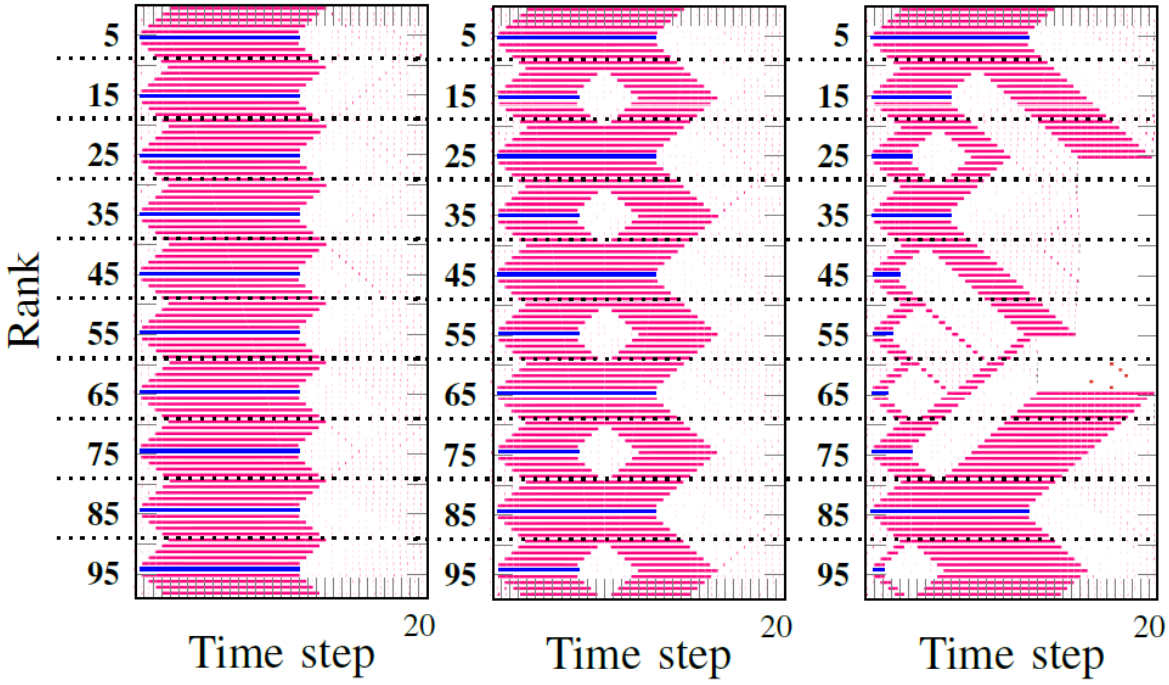




Idle waves interact nonlinearly

DOI: 10.1109/CLUSTER.2019.8890995

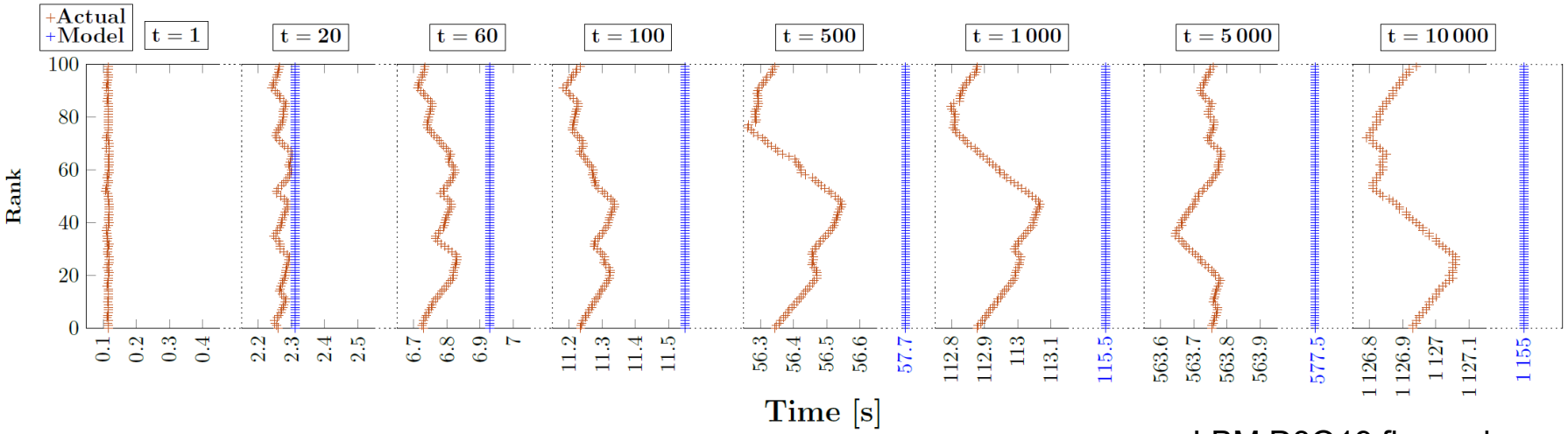
- A wave-like description cannot be based on a linear model
- Basis for noise-induced decay of idle waves



(a) equal (b) half (c) random

Spontaneous asynchronicity

There are always idle waves, and they have consequences…



LBM D3Q19 flow solver,
RRZE Emmy cluster,
10 cores per NUMA domain

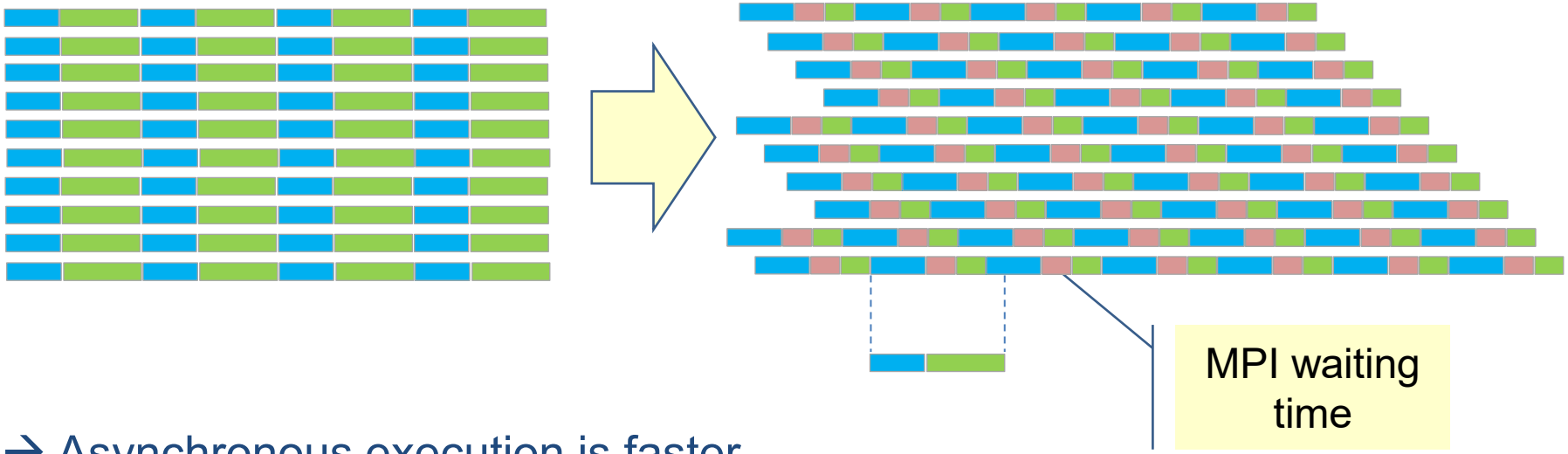
→ Opportunity for automatic communication overlap

→ "Spontaneous symmetry breaking"?

Spotaneous bottleneck overlapping

A two-phase MPI application

Ray tracer (core bound) + optical flow solver (memory b.)

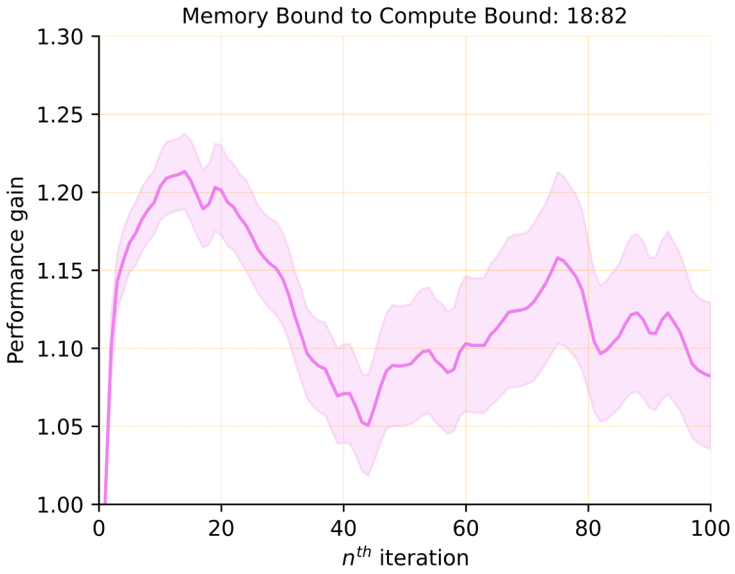


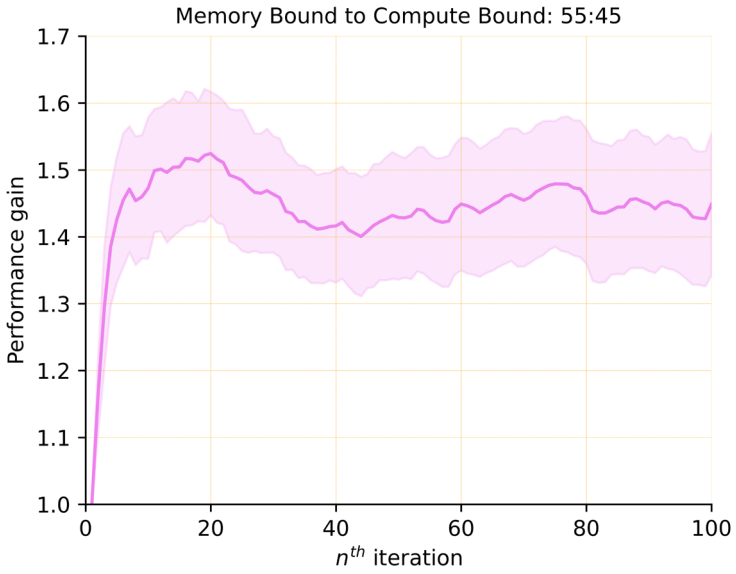
→ Asynchronous execution is faster

→ How does this happen?

A two-phase MPI application

Spontaneous evolution of code performance for different phase ratios





Data by Gayatri Manda

Asynchronicity can also be provoked by injecting noise!

Idle wave propagation and (de)synchronization phenomena

- A. Afzal, G. Hager, and G. Wellein: *Propagation and Decay of Injected One-Off Delays on Clusters: A Case Study.* Proc. 2019 IEEE International Conference on Cluster Computing (CLUSTER), Albuquerque, NM, September 23-26, 2019. DOI: 10.1109/CLUSTER.2019.8890995
- A. Afzal, G. Hager, and G. Wellein: *Desynchronization and Wave Pattern Formation in MPI-Parallel and Hybrid Memory-Bound Programs*. In: P. Sadayappan, B. Chamberlain, G. Juckeland, H. Ltaief (eds): High Performance Computing. ISC High Performance 2020. Lecture Notes in Computer Science, vol 12151. Springer, Cham. **Available with Open Access.** DOI: 10.1007/978-3-030-50743-5_20
- A. Afzal, G. Hager, and G. Wellein: *Analytic Modeling of Idle Waves in Parallel Programs: Communication, Cluster Topology, and Noise Impact*. ISC High Performance 2021 Digital, June 24 – July 2, 2021, Frankfurt, Germany. DOI: 10.1007/978-3-030-78713-4_19
- A. Afzal, G. Hager, and G. Wellein: *An analytic performance model for overlapping execution of memory-bound loop kernels on multicore CPUs*. Concurrency and Computation: Practice and Experience **34**(10), e6816 (2022). **Available with Open Access.** DOI: 10.1002/cpe.681
- A. Afzal, G. Hager, S. Markidis, and G. Wellein: *Making Applications Faster by Asynchronous Execution: Slowing Down Processes or Relaxing MPI Collectives*. Future Generation Computer Systems (2023), DOI: 10.1016/j.future.2023.06.017.


Ayesha Afzal

- A. Afzal, G. Hager, and G. Wellein: *The Role of Idle Waves, Desynchronization, and Bottleneck Evasion in the Performance of Parallel Programs*. IEEE Transactions on Parallel and Distributed Systems 34(2), 623-638 (2023), DOI: 10.1109/TPDS.2022.3221085.
- A. Afzal, G. Hager, and G. Wellein: *DisCostiC: Simulating MPI Applications Without Executing Code*. Poster at SC24

NHR FAU

Friedrich-Alexander-Universität
Erlangen-Nürnberg

Thank You.