Friedrich-Alexander-Universität
Erlangen-Nürnberg

The Role of Idle Waves in Modeling and Optimization of Parallel Programs

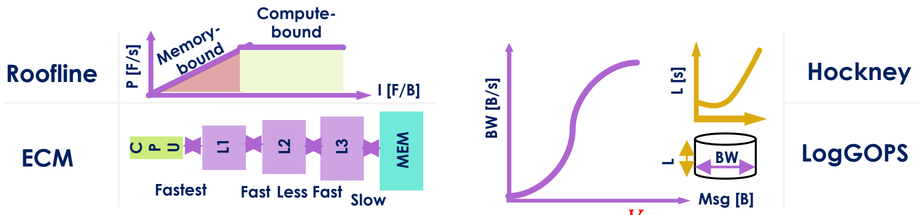
Ayesha Afzal[1,2], Georg Hager[1], Gerhard Wellein[1,2]

[1] Erlangen National High Performance Computing Center
[2] Department of Computer Science,
University of Erlangen-Nürnberg Germany

NHR PerfLab Seminar
April 26, 2022

Challenges of white-box performance modelling in HPC

Vision: white-box first-principle performance modelling



$\max(T_{BW}, T_{flops})$ $\lambda + \frac{V}{B}$

Process 0	T_{exec} Execution perf model	$T_{non-exec}$ Communication perf model	…
⋮			
Process N	T_{exec} Execution perf model	$T_{non-exec}$ Communication perf model	…

Accelerator Slowdown

$T_{comp} = sum(T_{exec}, T_{non-exec})$

Composite analytic performance model

$T_{dist} = sum(T_{exec}, T_{non-exec}) \pm T_{\delta}$

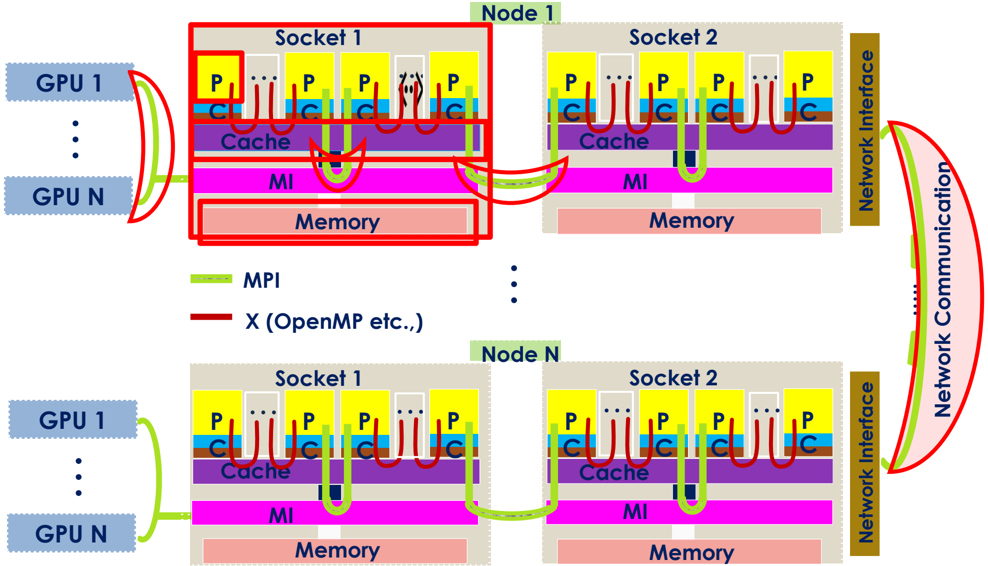
Distributed performance model

T_{δ}
(Out of) lock-step

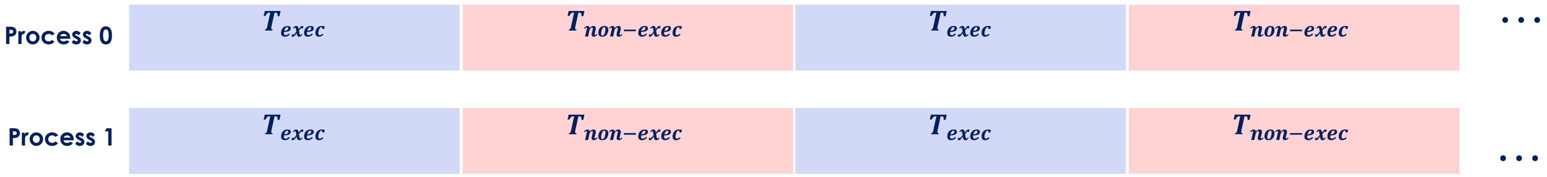
Why it's not a realistic model?

Why it can go in either direction?

It’s intricated (bottlenecks interact, systems are noisy, etc.)



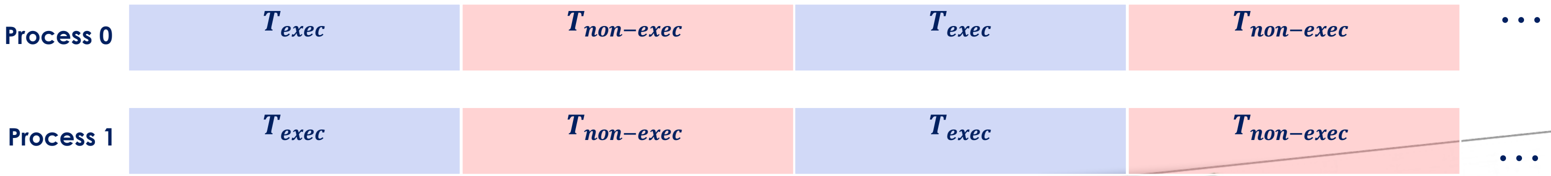
Motivation



"Lock-step" behavior at start

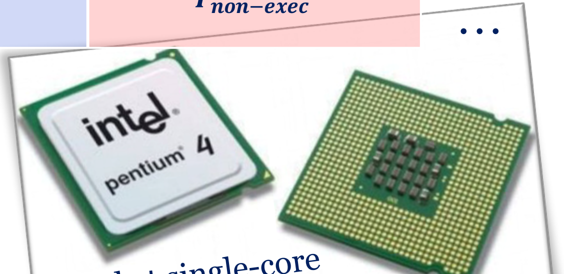
No network contention or load imbalance

Memory-bound MPI-parallel programs: timeline view



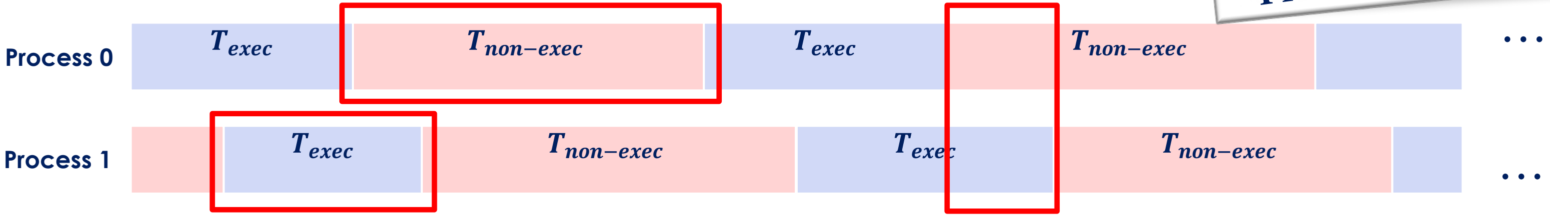
"Lock-step" behavior at start

No network contention or load imbalance



2 socket single-core "Prescott" node

Memory-bound MPI-parallel programs: timeline view

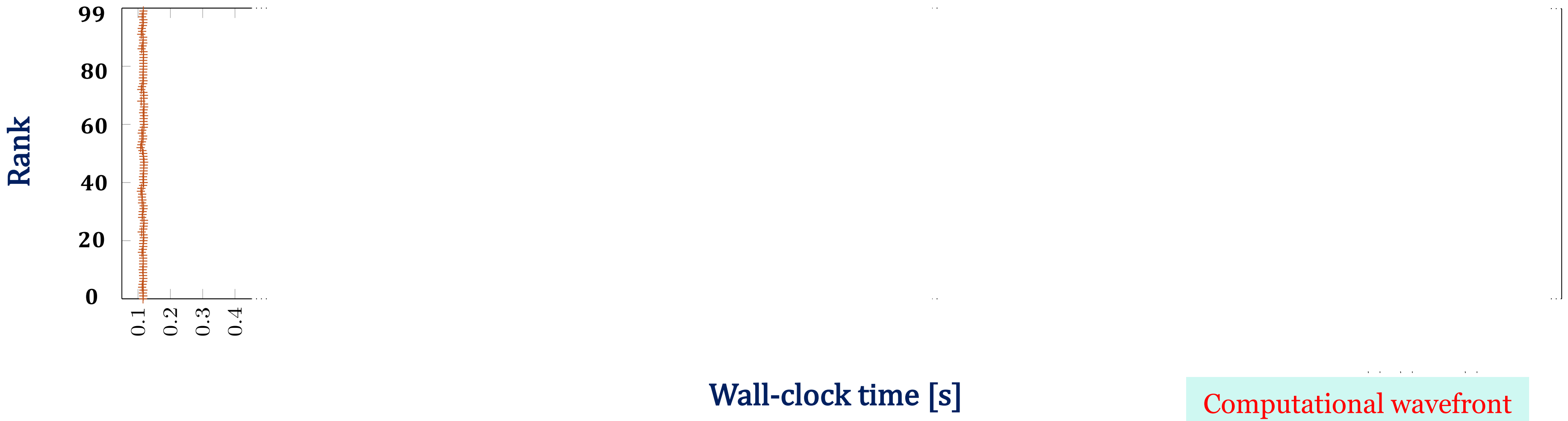


"Snap-in" desynchronization with communication overlap after some evolution

Motivating example (1)

MPI parallel Lattice Boltzmann fluid solver

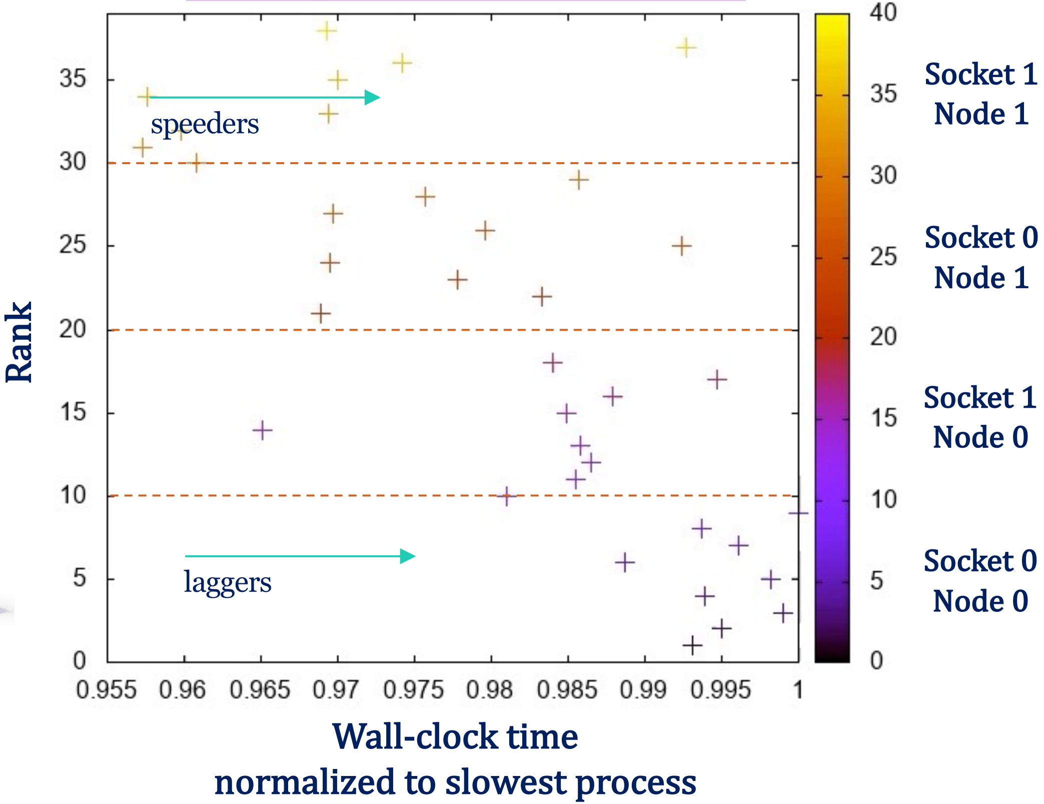
Time step 1

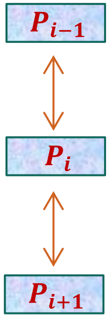


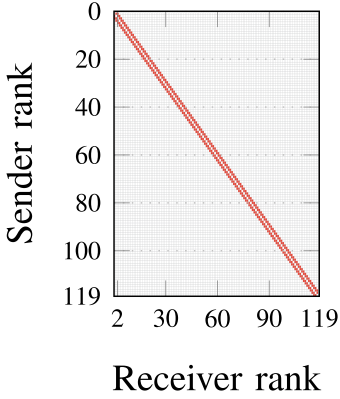
Computational wavefront

302^3 lattice cell, 8 GB data set, non-blocking, 1D domain decomposition, distance-1 communication, 10 Emmy@RRZE sockets @2.2 GHz

MPI parallel STREAM TRIAD





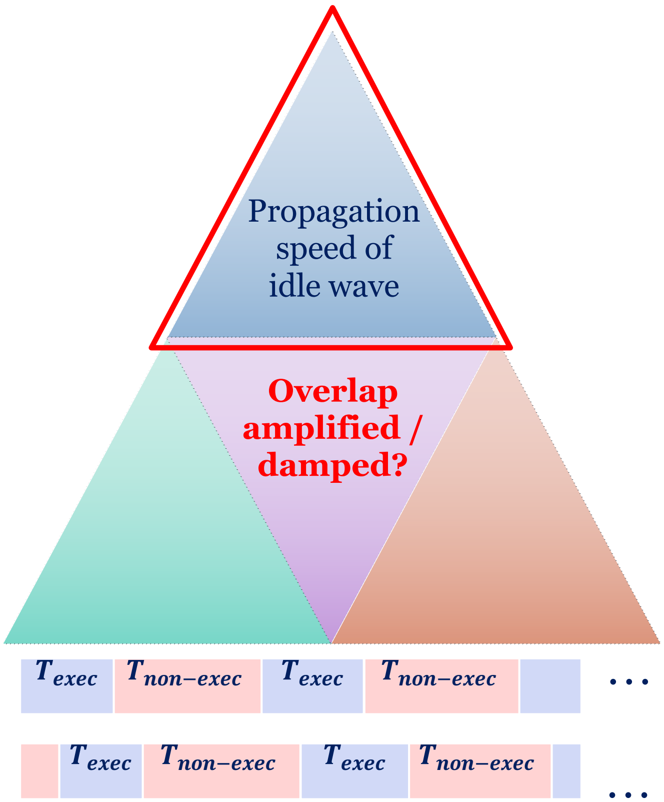


Spontaneous symmetry breaking, communication overlap, why?

Under what conditions?

4 Emmy@RRZE sockets @2.2 GHz, non-temporal stores, bi-dir, open chain, distance-1 communication

Lessons learned (1): Impact of idle wave on overlap

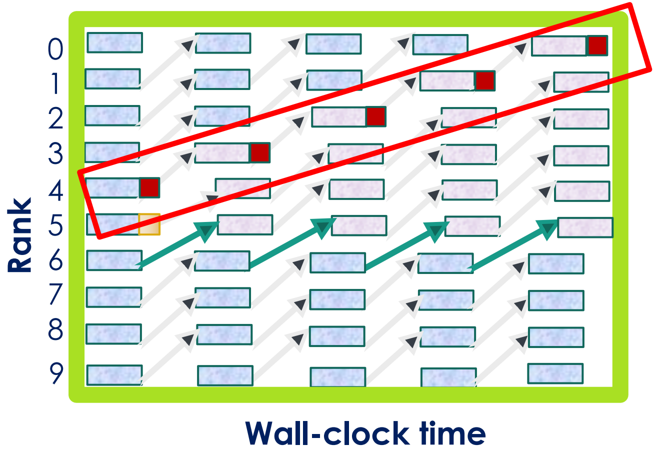


Idle wave: delay propagation across processes in each iteration

$$v_{min} = \kappa \times \sigma \times 1 \left[\frac{rank}{iter}\right] \times \frac{1}{T_{comp} + T_{comm}} \left[\frac{iter}{s}\right]$$

$$\kappa = \frac{j(j+1)}{2} \text{ or } (j + \lfloor i \rfloor) or\ j$$

$j\,/i$: longest /shorter-distance partner



resource-scalable MPI programs

DOI: 10.1103/PhysRevE.91.013306

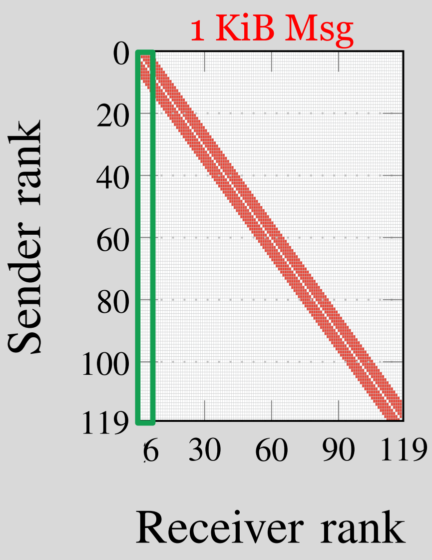
Communication pattern and concurrency: compact

```
1: while d ≤ dims do
2:     while dir ≤ bi do
3:         MPI_Isend ‡
4:         MPI_Irecv §
5:     end while
6:     MPI_Waitall
7: end while
```

```
1: while dir ≤ bi do
2:     while d ≤ dims do
3:         MPI_Isend ‡
4:         MPI_Irecv §
5:     end while
6:     MPI_Waitall
7: end while
```

Dimensionless κ

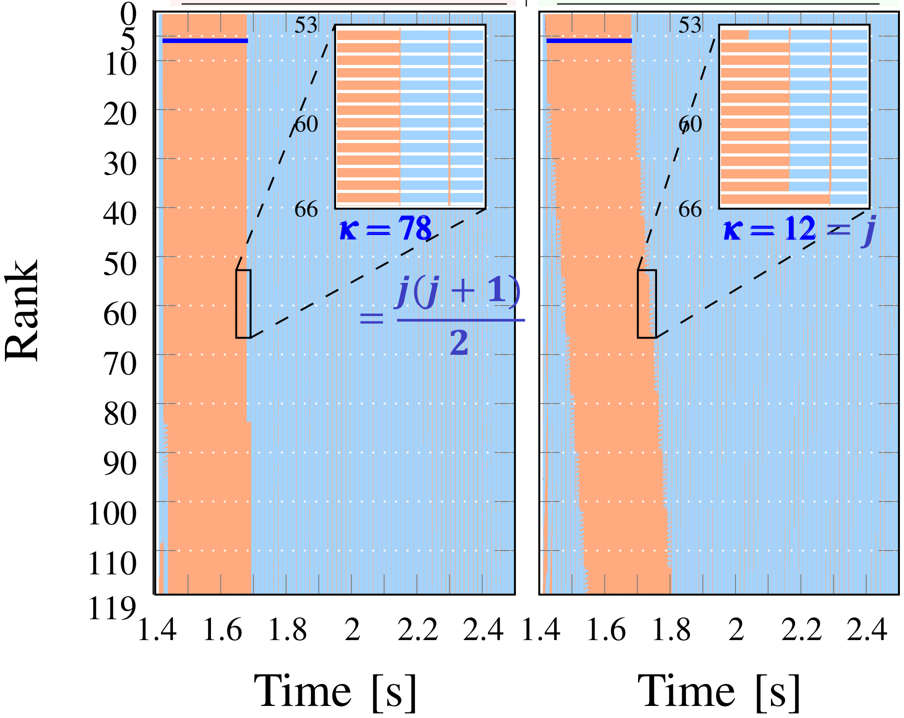
distance in processes travelled in one time step by the idle wave



Communication pattern and concurrency: compact

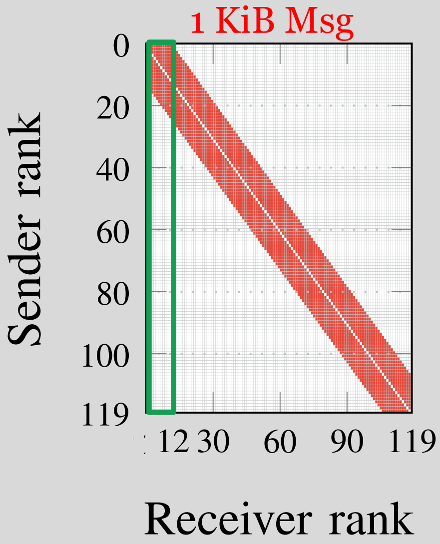
```
while d ≤ dims do
    while dir ≤ bi do
        MPI_Isend ‡
        MPI_Irecv §
    end while
    MPI_Waitall
end while
```

```
while dir ≤ bi do
    while d ≤ dims do
        MPI_Isend ‡
        MPI_Irecv §
    end while
    MPI_Waitall
end while
```



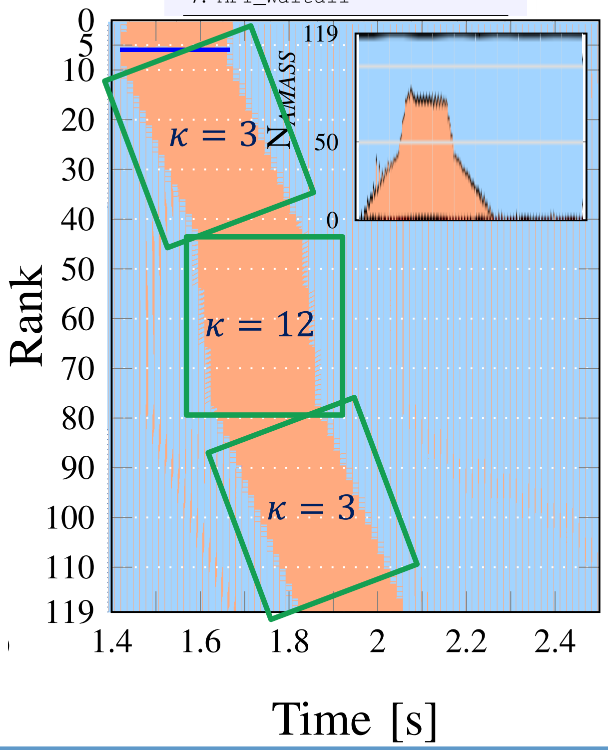
Dimensionless κ

distance in processes travelled in one time step by the idle wave

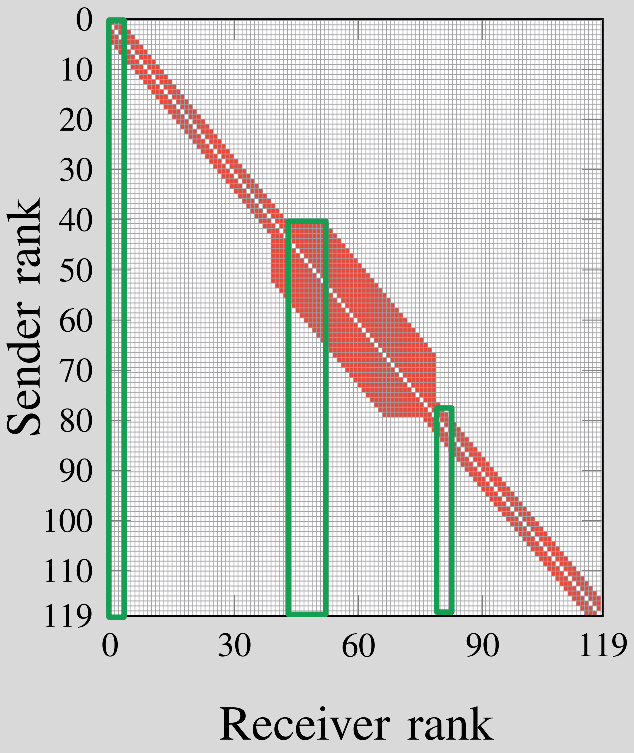


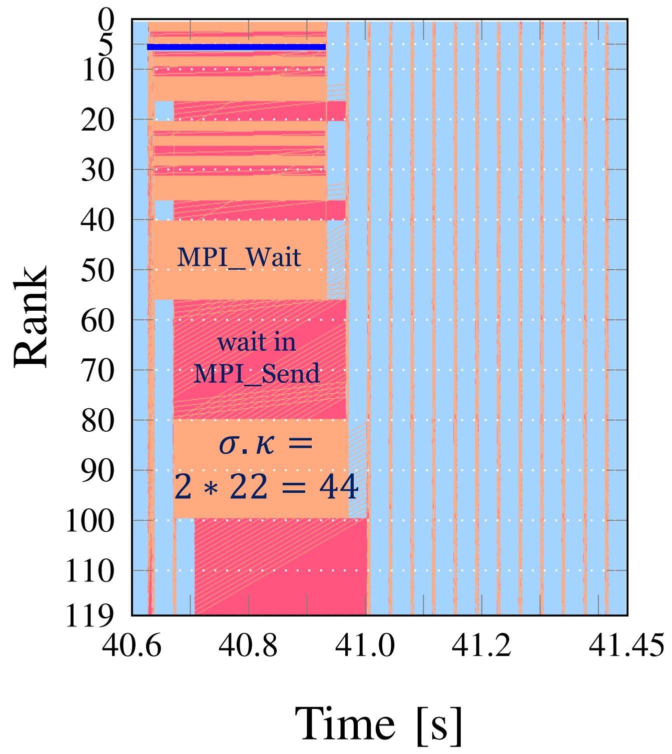
Communication pattern and concurrency: heterogeneous

```
1: while d ≤ dims do
2:     while dir ≤ bi do
3:         MPI_Isend ‡
4:         MPI_Irecv §
5:     end while
6: end while
7: MPI_Waitall
```



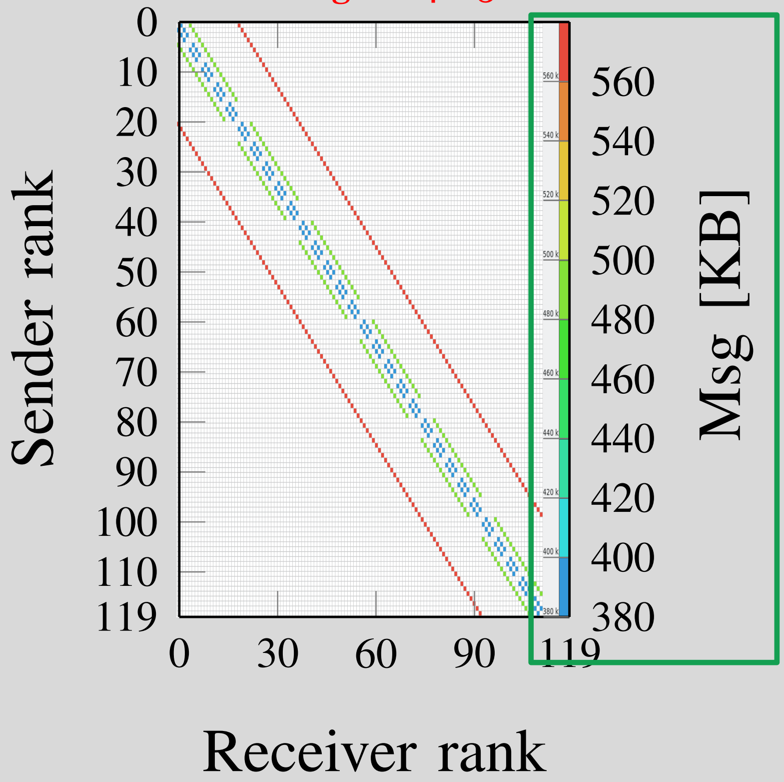
Refraction effect



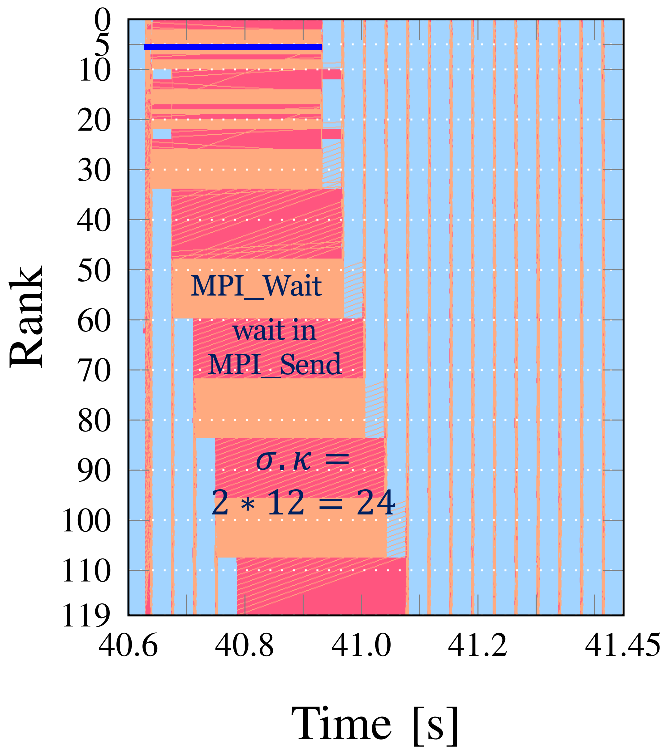


One MPI process per ccNUMA

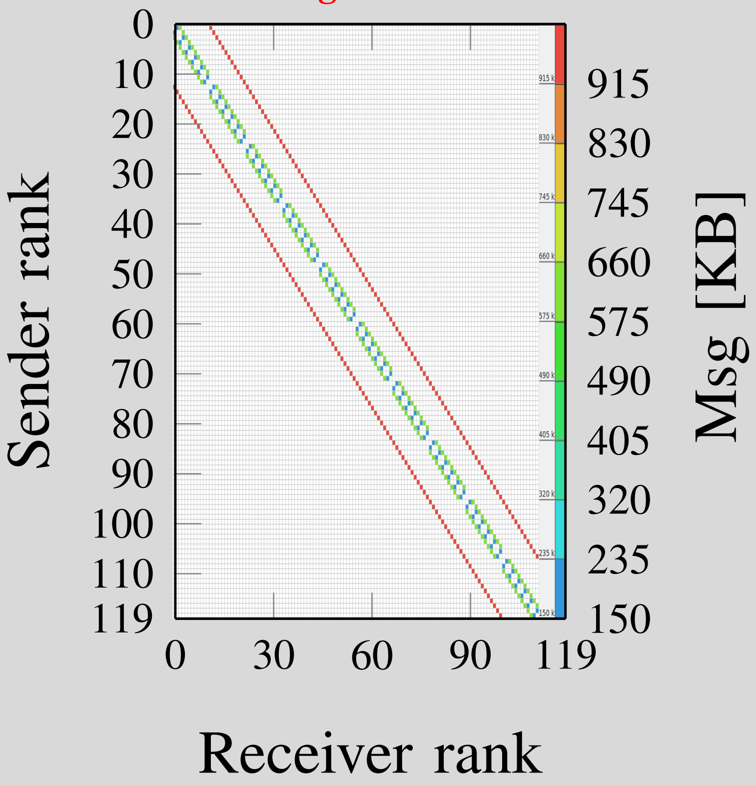
Process grid: 4 x 5 x 6



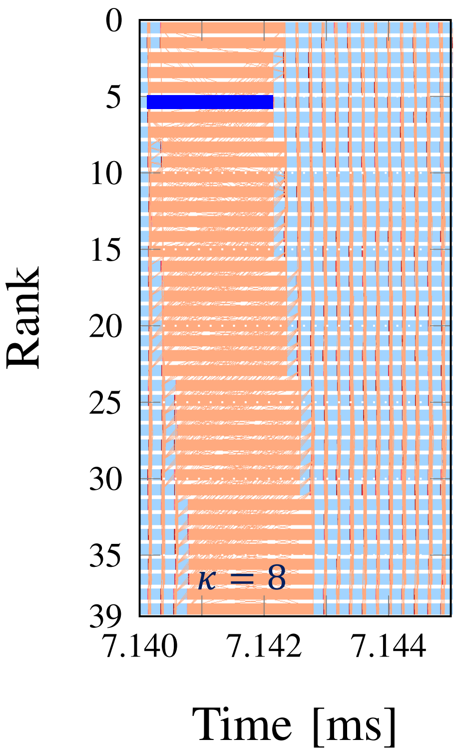
Idle wave propagation: 3D jacobi smoother



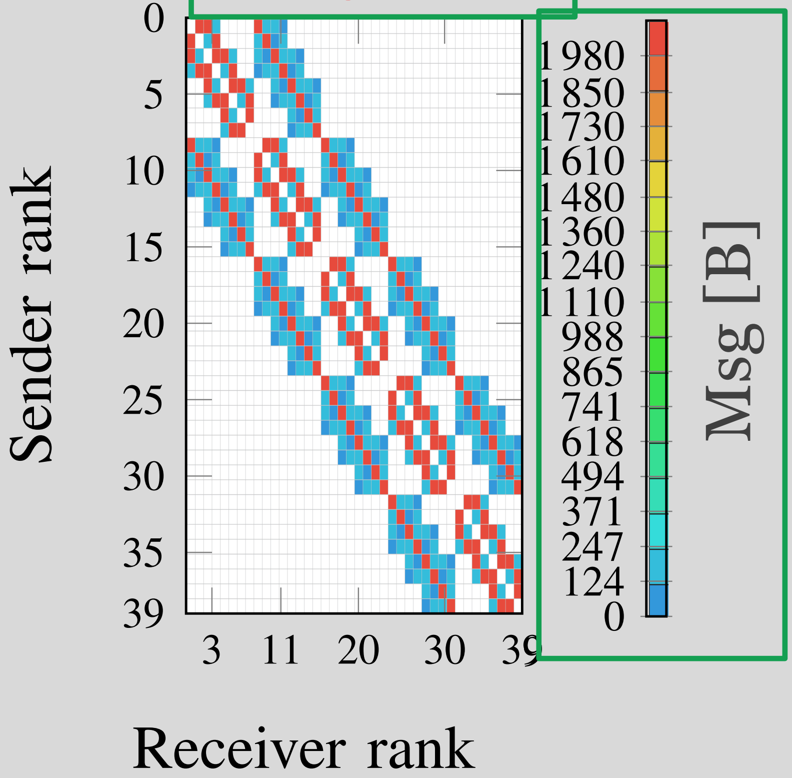
One MPI process per ccNUMA
Process grid: 2 x 6 x 10



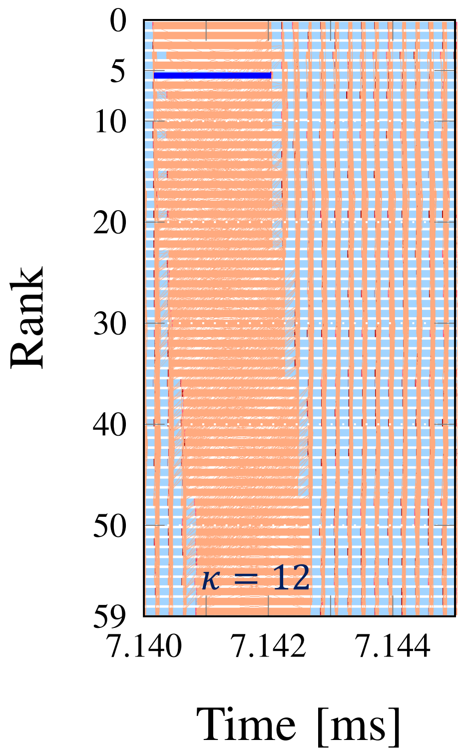
Idle wave propagation: sparse MVM with HPCG matrix

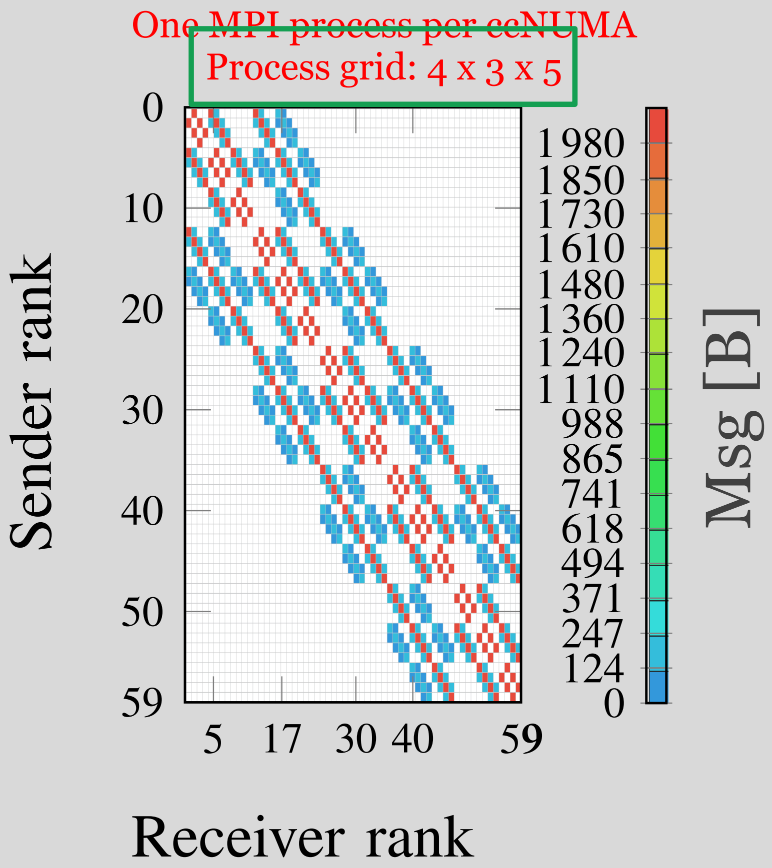


One MPI process per ccNUMA
Process grid: 2 x 4 x 5

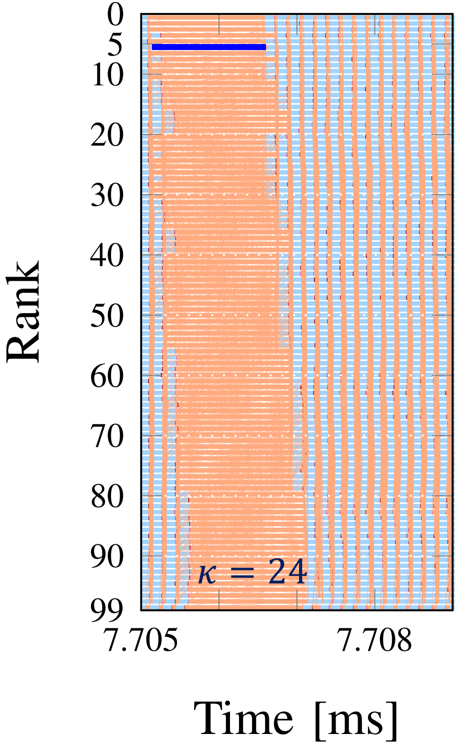


Idle wave propagation: sparse MVM with HPCG matrix



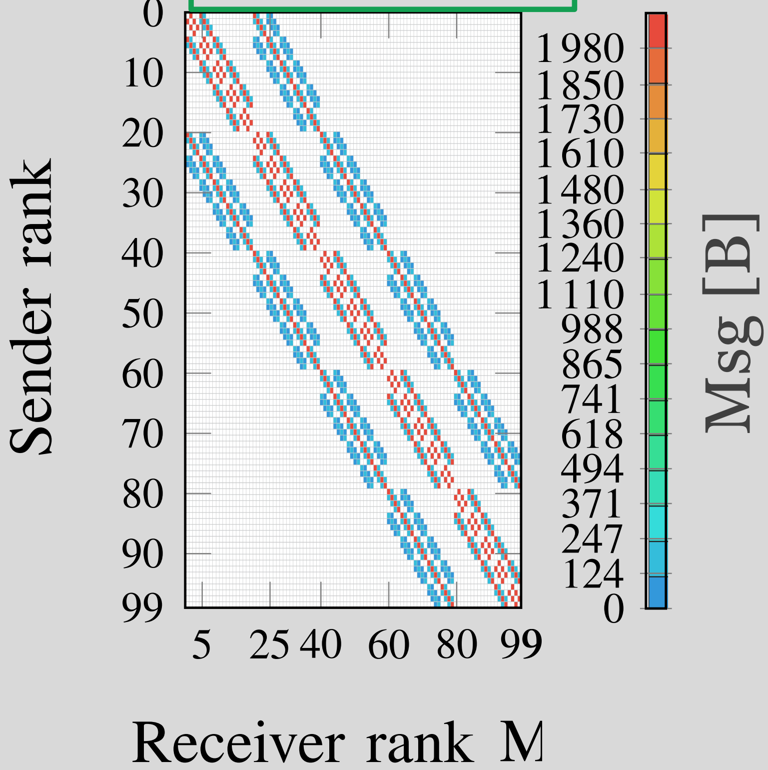


Idle wave propagation: sparse MVM with HPCG matrix

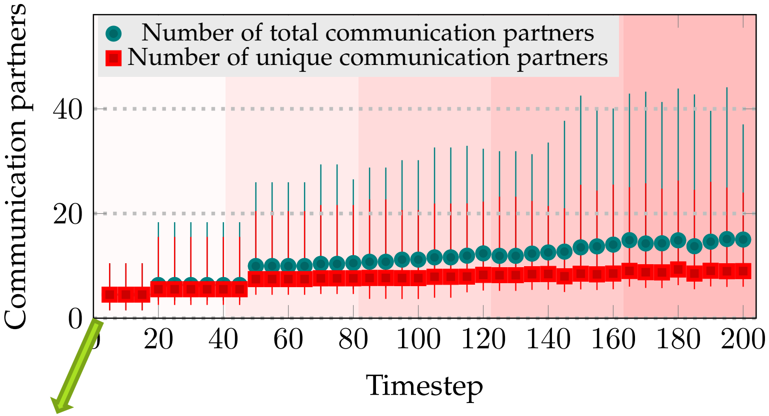


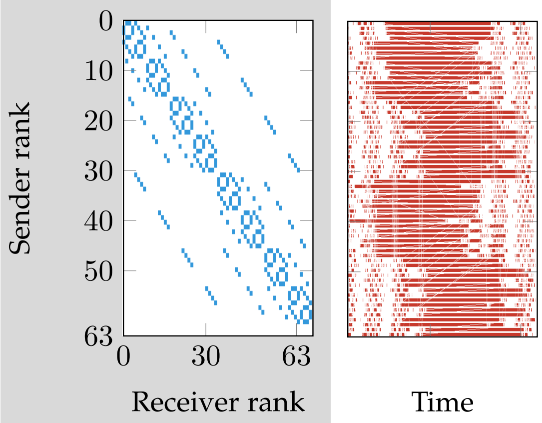
One MPI process per ccNUMA

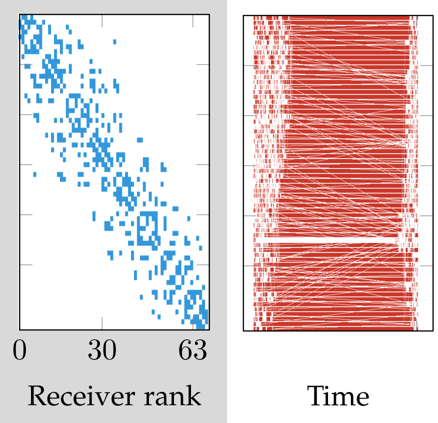
Process grid: 4 x 5 x 5



Idle wave propagation: Adaptive Mesh Refinement - miniAMR





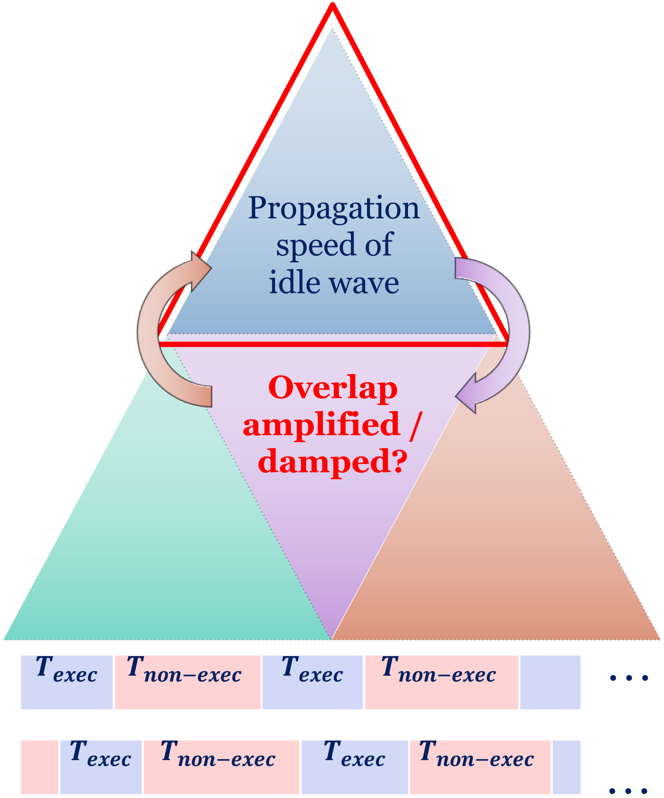


Lessons learned: Impact of idle wave on overlap

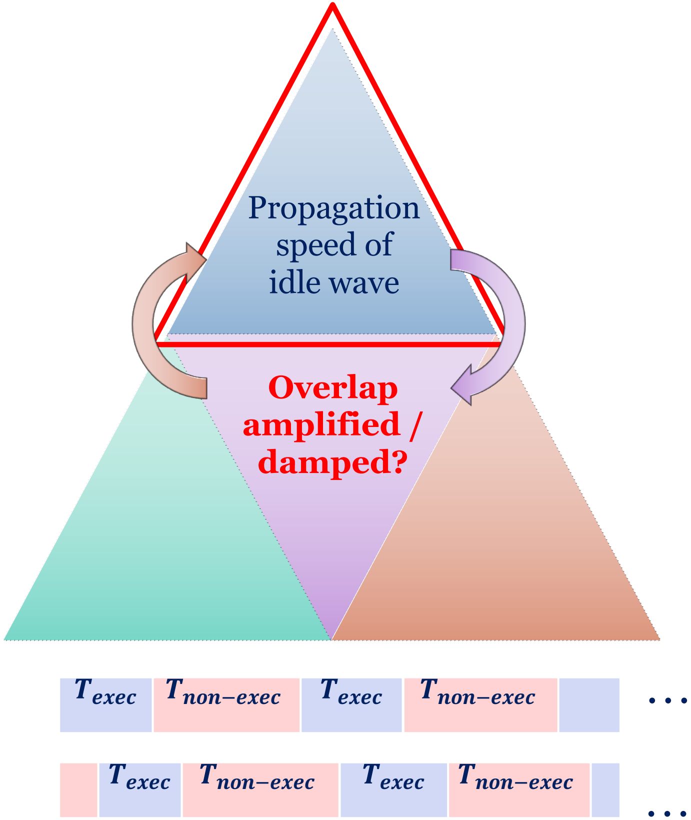
Afzal et al.

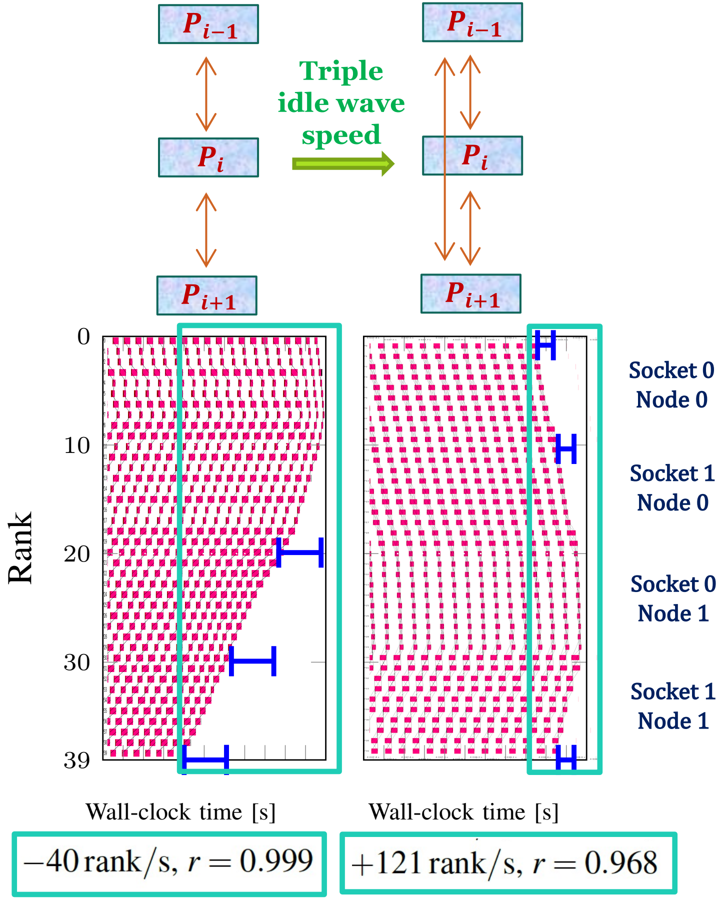
IEEE Cluster 2019

DOI:10.1109/CLUSTER.2019.8890995



Lessons learned: Impact of idle wave on overlap



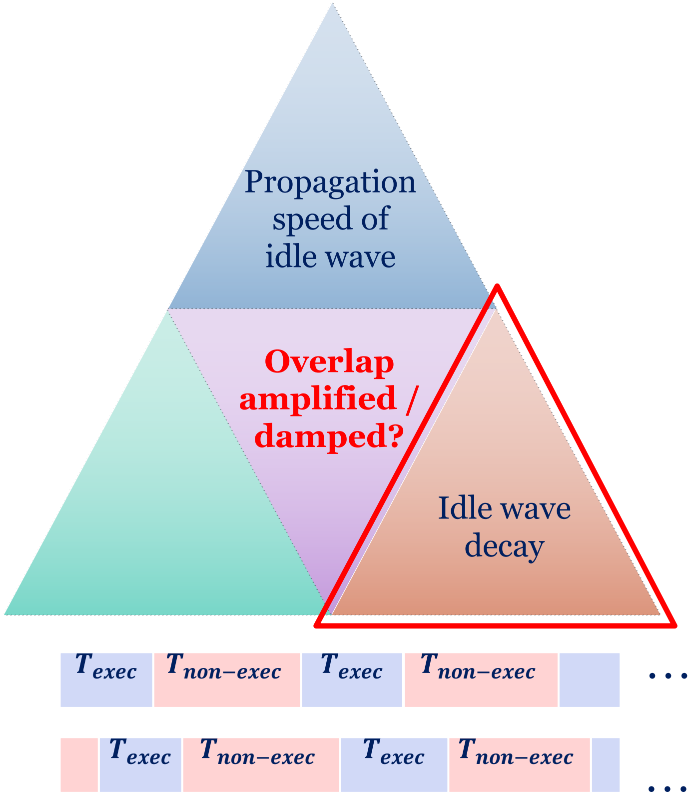


Lessons learned: Impact of idle wave on overlap

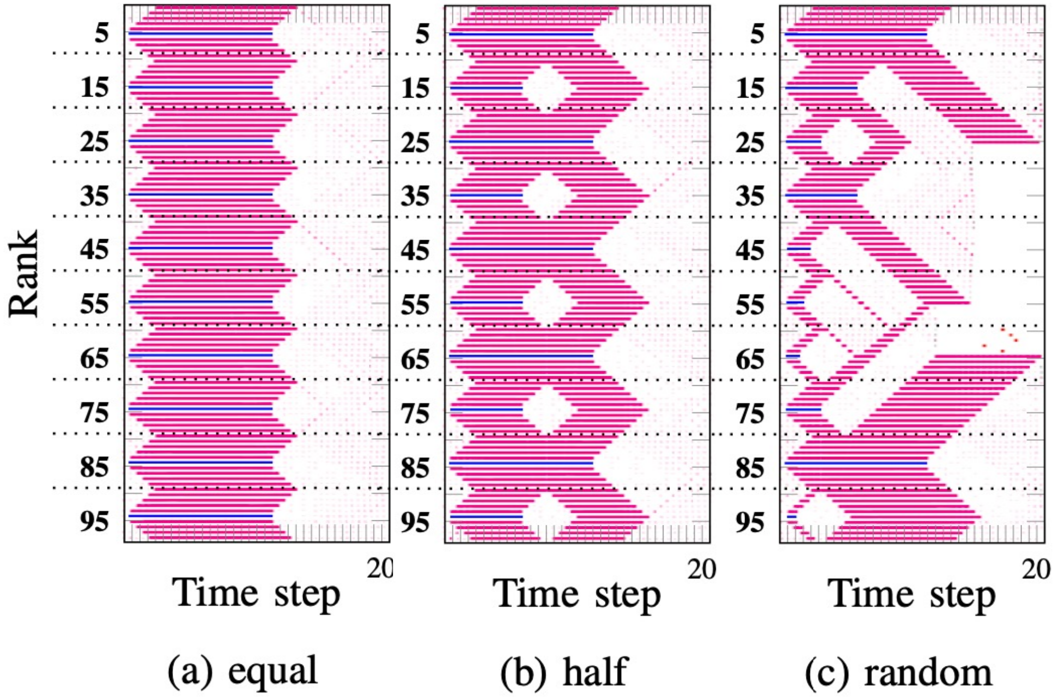
Afzal et al.

IEEE Cluster 2019

DOI:10.1109/CLUSTER.2019.8890995



Non-linear interactions of idle wave

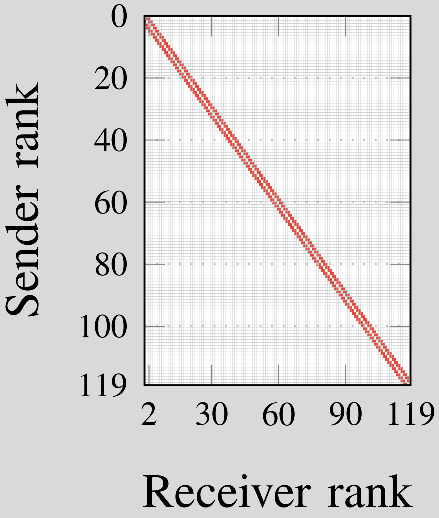


Non-linear interactions of idle wave with system topology

1 KiB Msg

Emmy	SuperMUC-NG	Hawk
2 x 1 x 10	2 x 1 x 24	2 x 4 x 16
(12 domains)	(5 domains)	(30 domains)

Single Node:
sockets per node x
ccNUMA domain per socket x
physical cores per domain



Lessons learned: Impact of idle wave on overlap

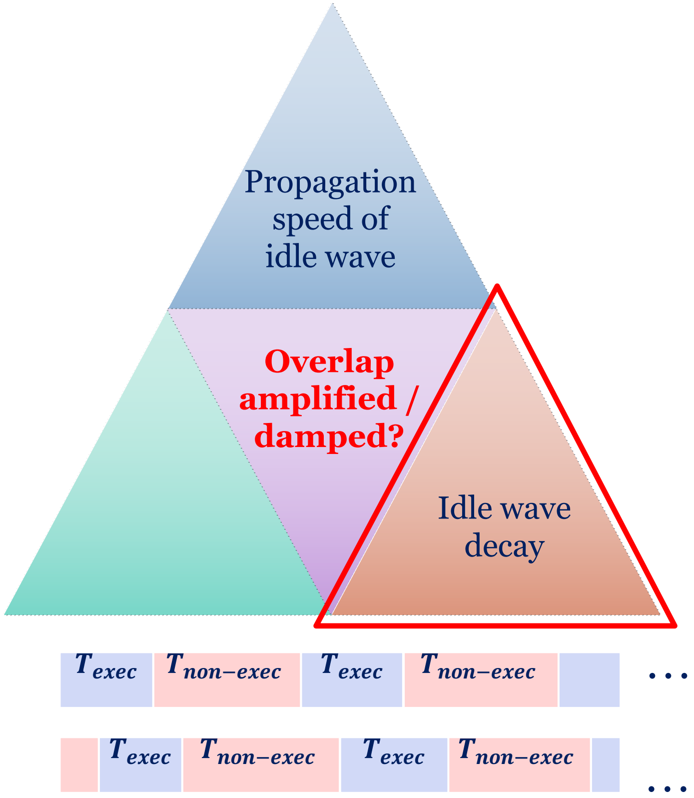
Afzal et al.

IEEE Cluster 2019

DOI:10.1109/CLUSTER.2019.8890995

ISC HPC 2021

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



Lessons learned: Impact of idle wave on overlap

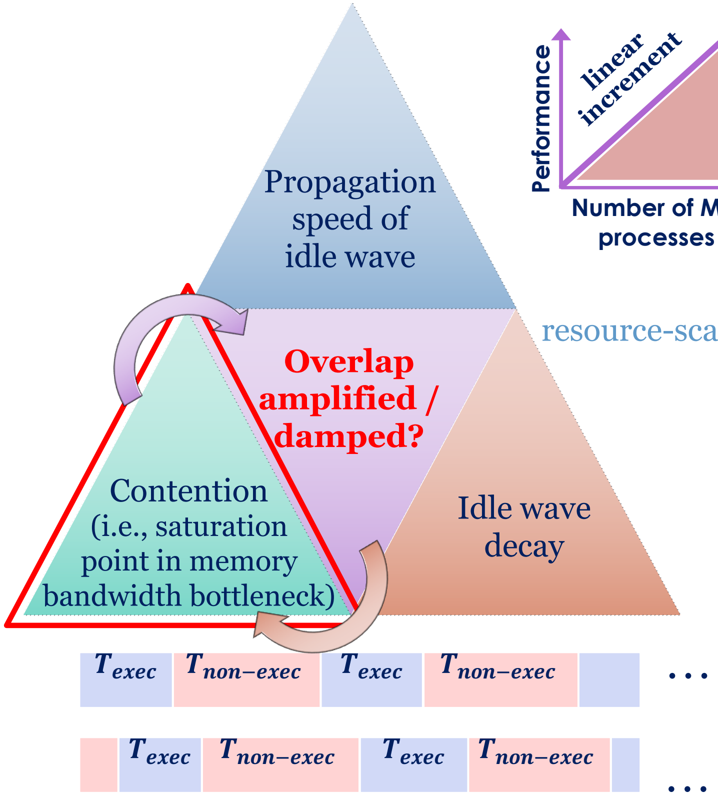
Afzal et al.

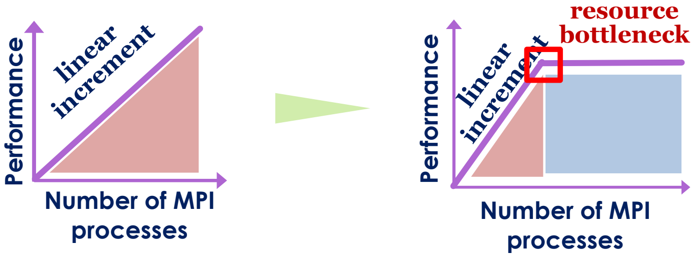
IEEE Cluster 2019

DOI:10.1109/CLUSTER.2019.8890995

ISC HPC 2021

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





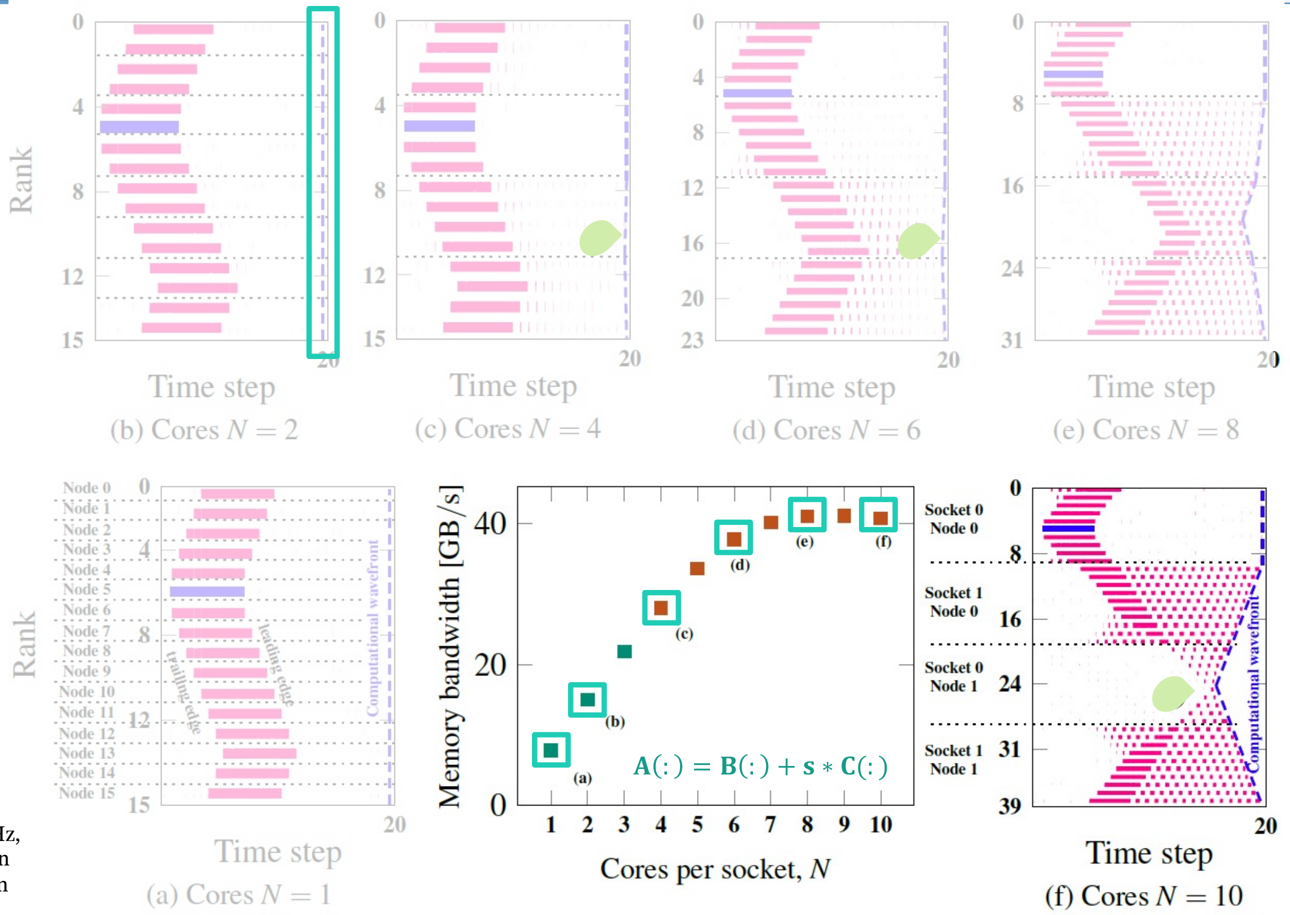
resource-scalable

memory-bandwidth bottleneck

Possible bottlenecks

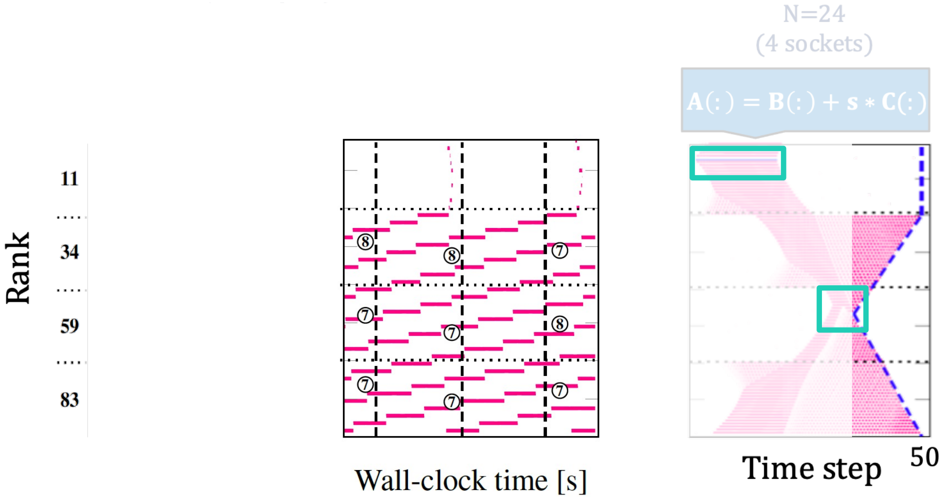
- **memory**
- cache
- on-/inter-chip network
- link b/w host & accelerator

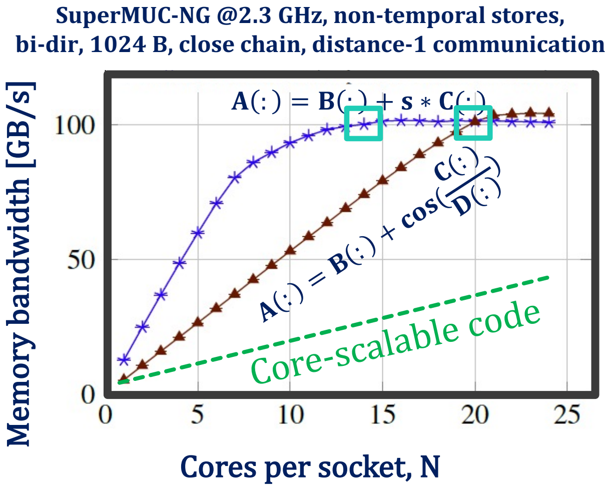
Scalable versus contented processes

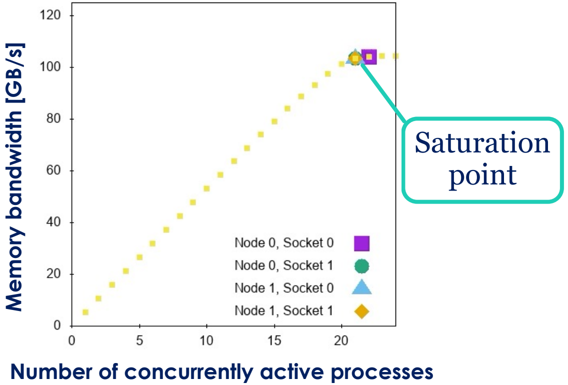


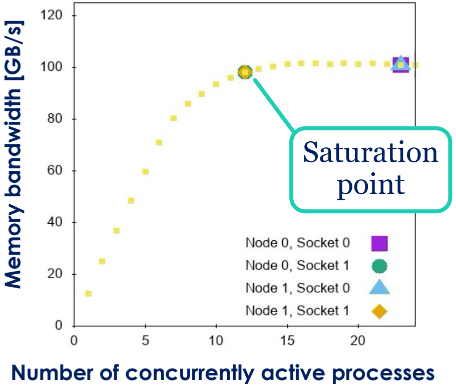
4 Emmy 10-core sockets @2.2 GHz, non-temporal stores, bi-dir, open chain, distance-1 communication

Scalable versus contented processes









Scalable processes ➡ No overlap

Strongly contented processes ➡ Maximum overlap

Afzal et al.

IEEE Cluster 2019

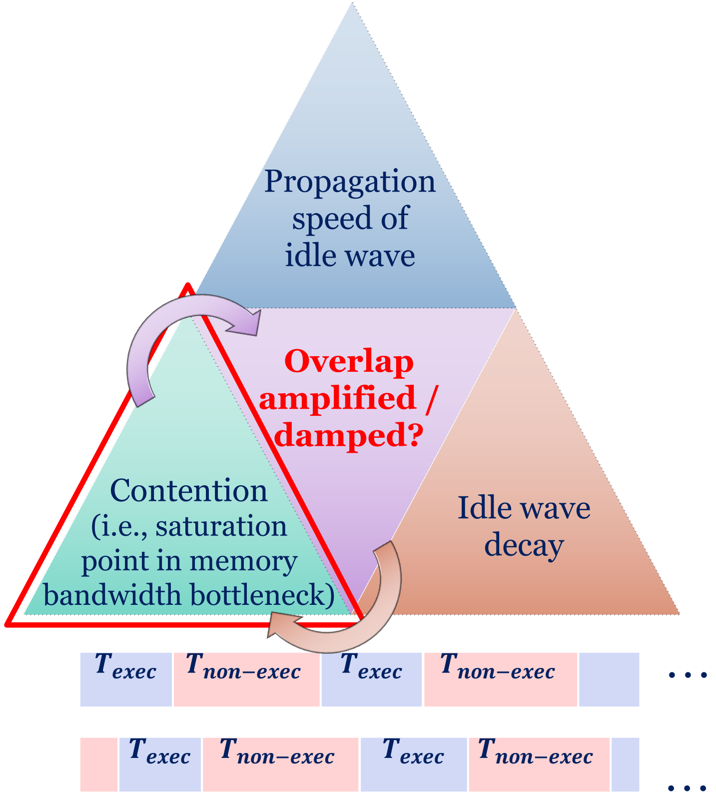
DOI:10.1109/CLUSTER.2019.8890995

ISC HPC 2021

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

ISC HPC 2020

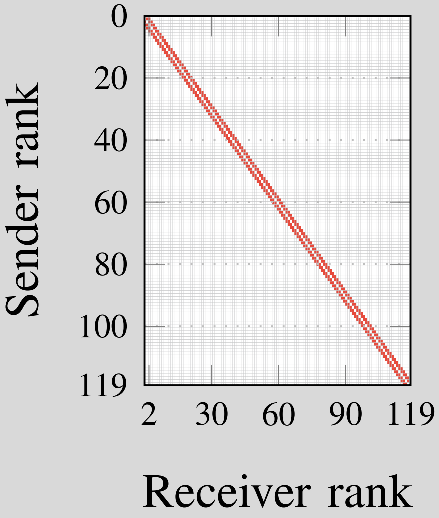
DOI: 10.1007/978-3-030-50743-5_20



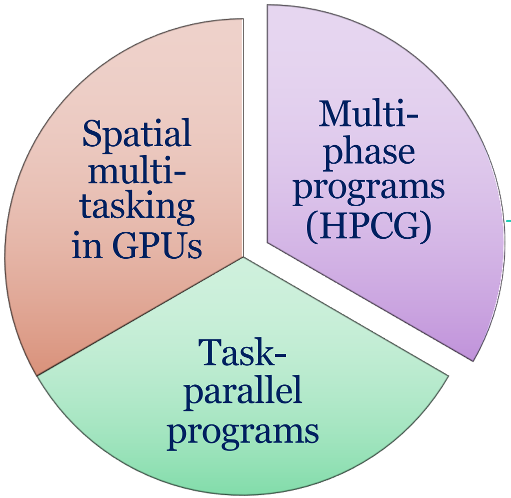
Lessons learned: Scope

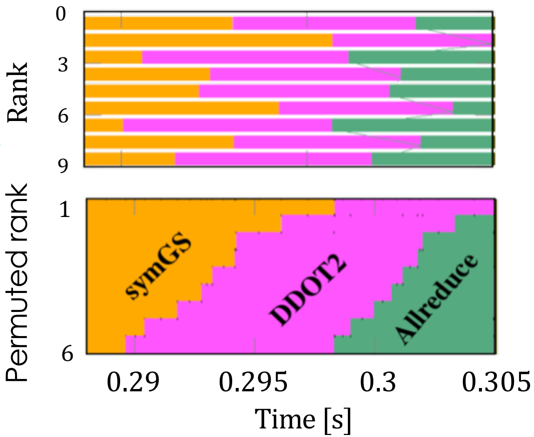
Algorithmic dependency: some collectives can be permeable to idle waves

Intel MPI 19.1



Lessons learned: Scope





Afzal et al.

CCPE 2022

DOI: 10.1002/cpe.6816

Performance: MPI parallel Sparse Matrix-Vector Multiplication

Listing 1: CRS based SPMVM kernel

```
double :: valA[nnz], b[nr], x[nr] ;
int :: colIdxA[nnz], rowPtrA[nr + 1], tmp ;
for row = 0 : nr − 1 do
    tmp = 0.0 ;
    for idx = rowPtrA[row] : rowPtrA[row + 1] − 1 do
        tmp += valA[idx] ∗ x[colIdxA [idx]] ;
    end for
    b[row] += tmp ;
end for
```

Listing 2: SPLIT-WAIT mode

```
while iter ≤ nIters do
    MPI_Irecv ;
    MPI_Isend ;
    local_spMVM (A, x, b) ;
    MPI_Wait ; ⋆
    remote_spMVM (A, x, b) ;
    MPI_Barrier ;
    swap (b, x) ;
end while
```

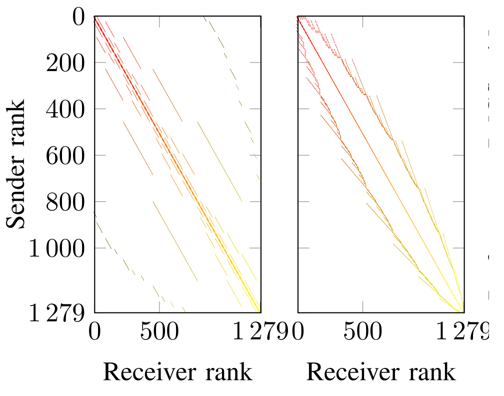
Listing 3: NON-SPLIT mode

```
while iter ≤ nIters do
    MPI_Irecv ;
    MPI_Isend ;
    MPI_Wait ; ⋆
    local_spMVM (A, x, b) ;
    remote_spMVM (A, x, b) ;
    MPI_Barrier ;
    swap (b, x) ;
end while
```

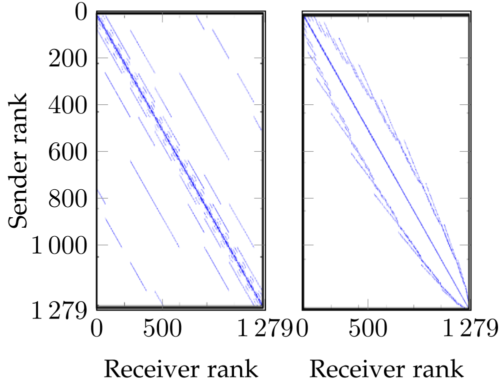
⋆ Two MPI_Wait routines wait for both MPI receive and send requests to complete.

Matrix-order	Bandwidth	$n_{\text{electrons}} - n_{\text{sites}} - n_{\text{phonons}}$ §	$n_r = n_c$ ⋆	n_{nz} ⋆	n_{nzr} ‡	Size [GB] ¶
HHQ-large-*pe*	high	3 − 8 − 10	60988928	889816368	13	10.9
HHQ-large-*ep*	low	3 − 8 − 10	60988928	889816368	13	10.9
HHQ-small-*pe*	high	6 − 6 − 15	6201600	92527872	15	1.14
HHQ-small-*ep*	low	6 − 6 − 15	6201600	92527872	15	1.14

Phase vs. Rank-order	96-pe	144-pe	240-pe	480-pe	720-pe	960-pe	1296-pe	96-ep	144-ep	240-ep	480-ep	720-ep	960-ep	1296-ep
Exec median [ms]	53.5	35.74	18.51	9.05	6.53	5.06	3.85	48.47	30.87	17.57	8.23	5.7	4.54	3.03
Comm median [ms]	29.48	24.56	16.17	14.75	11.28	9.78	7.99	16.17	15.01	14.41	12.17	8.96	6.62	6.13
CER median	0.55	0.69	0.87	1.63	1.73	1.93	2.08	0.33	0.49	0.82	1.48	1.57	1.46	2.02

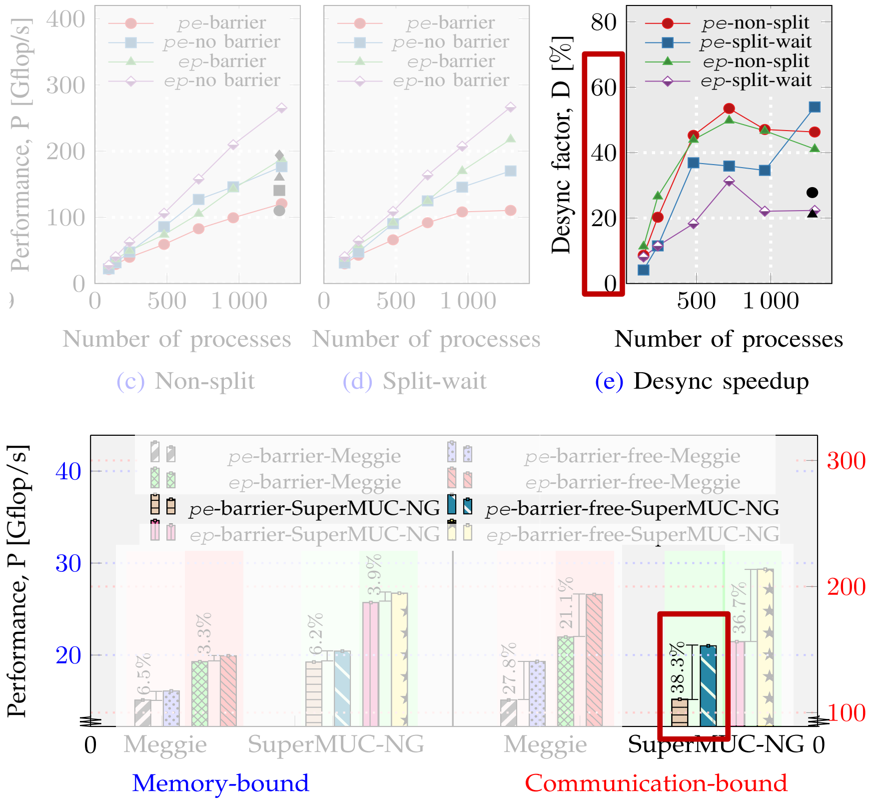


(a) pe order (b) ep order



(a) pe order (b) ep order

Performance: MPI parallel Sparse Matrix-Vector Multiplication



LARGE COMMUNICATION OVERHEAD

High matrix bandwidth (pe)

SLOW IDLE WAVE

Low matrix bandwidth (ep)

SLOW SYNCHRONIZED BASELINE

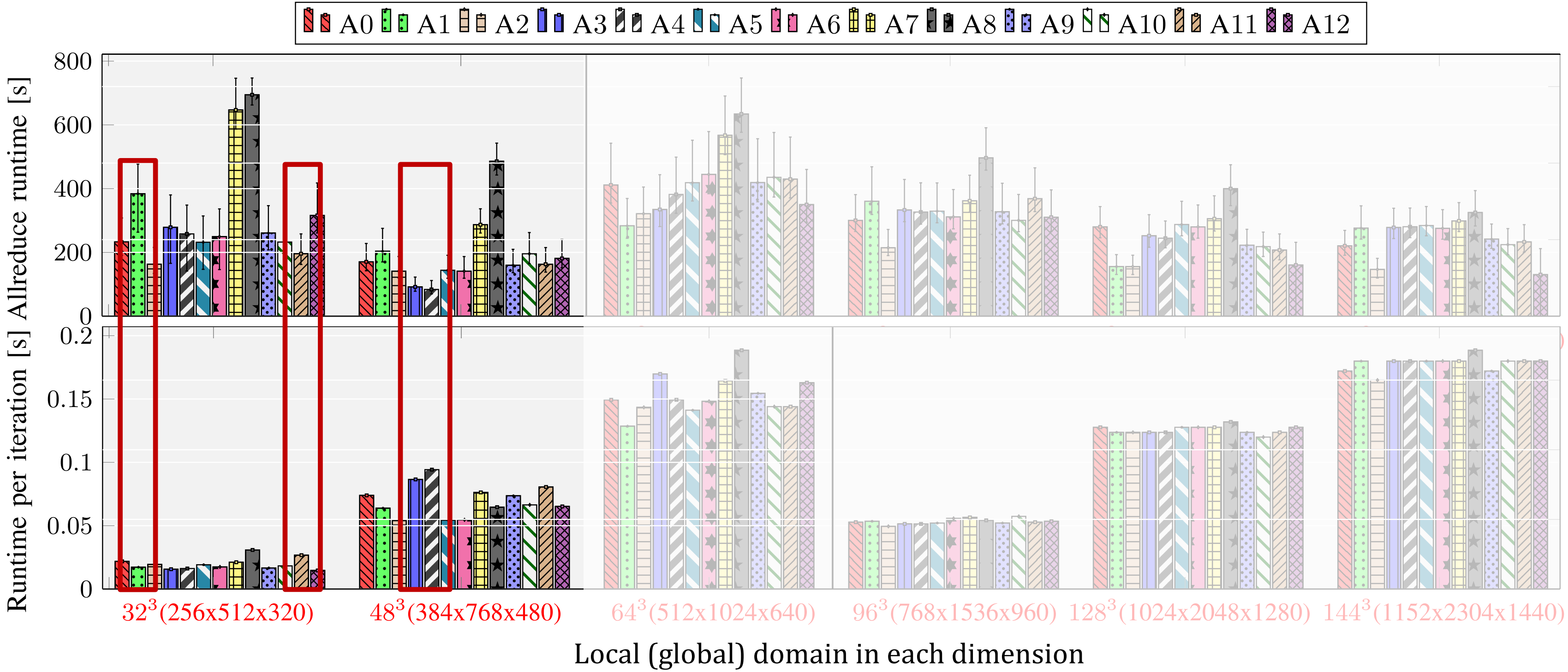
Non-split communication scheme

Higher speedup with automatic overlap

$$D[\%] = \frac{P_{barrier\_free} - P_{barrier}}{|P_{barrier}|} \times 100$$

Performance: MPI parallel High Performance Conjugate Gradient

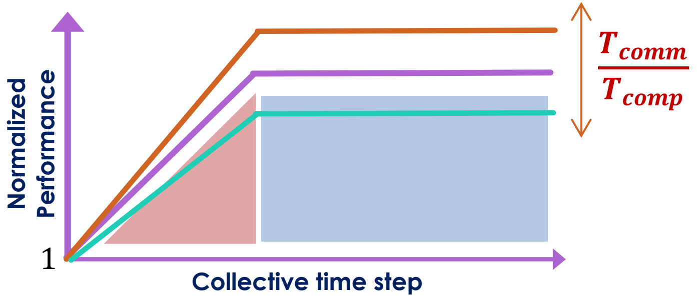
Better speed-up with less-synchronizing collective variant even if it not the most efficient implementation



Large amount of time spent in the MPI library is not necessarily harmful if overlapping useful work

Pseudo implementation of parallel LBM

```
while iter ≤ nIters do
    stream_collide_update (lattice, u_lid, omega);
    set_boundary_condition (u_lid);
    MPI_Isend ; *
    MPI_Irecv ; *
    MPI_Wait ;
    ghost_cells_update ();
    if ((iter % collective_step) == 0) then
        MPI_Allreduce ;
     end if
     swap (local_src_lattice, local_dst_lattice) ;
end while
```

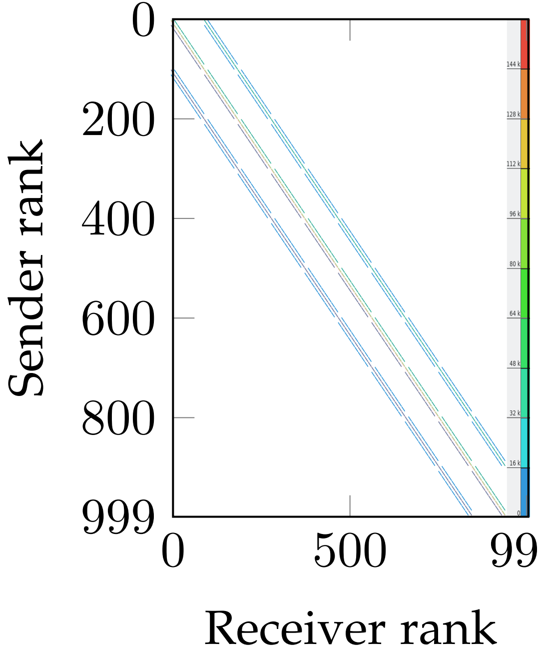


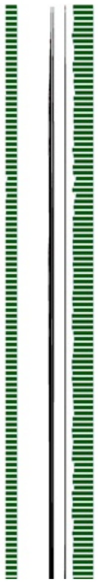
NOISE IS NOT DETRIMENTAL

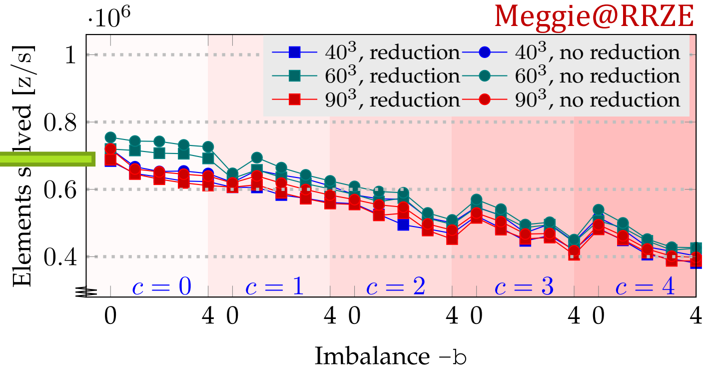
High collective frequency

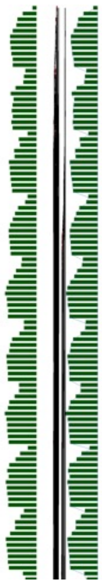
Speedup with higher automatic overlap

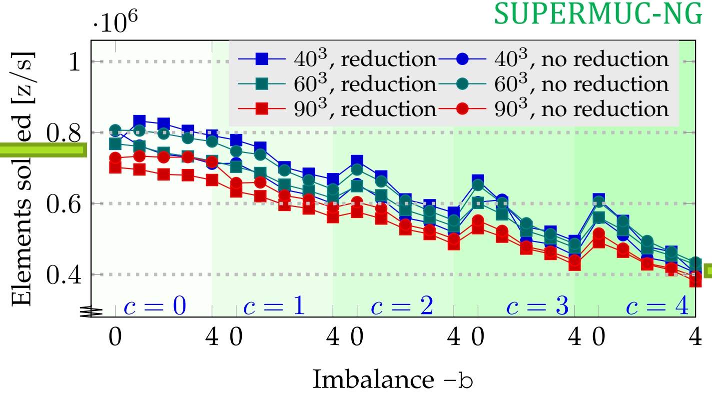
Performance: MPI parallel LULESH proxy application











Overlap effect is swamped by dominating laggers ➔ no benefit

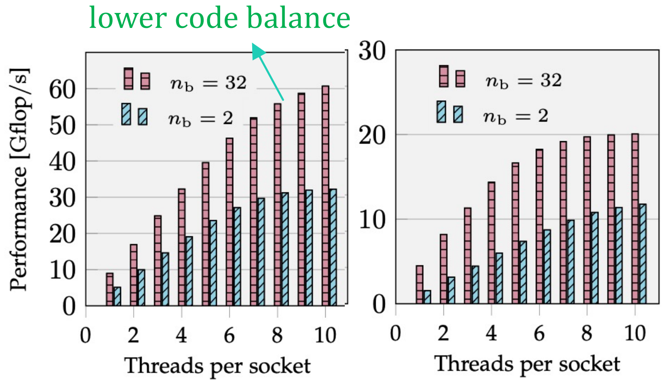
Performance: Hybrid parallel Chebychev Filter Diagonalization

NON-SPLIT mode
for k $= 0 : \frac{n_s}{n_b} - 1$ **do**
for p $= 0 : n_p - 1$ **do**
swap$(\vec{W}, \vec{U})$;
comm_init_finalize$(\vec{U})$;
$\vec{W} \leftarrow 2(\alpha \vec{H} + \beta)\vec{U} - \vec{W}$;
$\vec{\eta}_p \leftarrow \langle \vec{W}, \vec{U} \rangle$;
$\vec{\mu}_p \leftarrow \langle \vec{U}, \vec{U} \rangle$;
$\vec{X} \leftarrow \vec{X} + g_p c_p \vec{W}$;
end for
end for

kernel

SPLIT-WAIT mode
for k $= 0 : \frac{n_s}{n_b} - 1$ **do**
for p $= 0 : n_p - 1$ **do**
swap$(\vec{W}, \vec{U})$;
comm_init$(\vec{U})$;
local_kernel ;
MPI_wait() ;
remote_kernel ;
end for
end for

PIPELINE mode
for p $= 0 : n_p - 1$ **do**
swap$(\vec{W}, \vec{U})$;
comm_init$(\vec{U}_0)$;
MPI_wait() ;
for k $= 0 : \frac{n_s}{n_b} - 2$ **do**
comm_init$(\vec{U}_1)$;
kernel$_k$() ;
MPI_wait() ;
end for
kernel$_{\frac{n_s}{n_b}-1}$() ;
end for

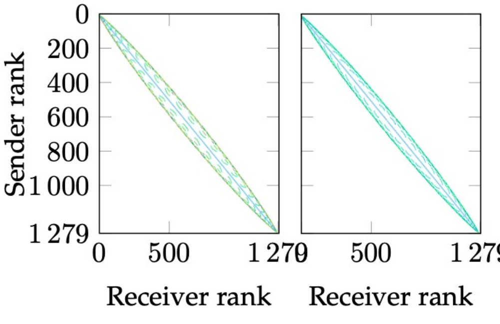


(a) TOPI-EHN
(b) SPIN26

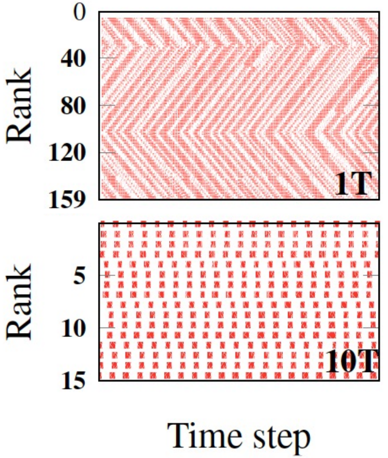
Matrix	Traits[§]	Data-type	$n_r = n_c$	n_{nz}	n_{nzpr}	Size [GB][‡]
TOPI-ENH	m_x-m_y-m_z-128-128-64	complex double	268435456	3487563776	13	56.9[‡]
SPIN26*	26-13-1	double	10400600	145608400	14	1.8
SPIN28*	28-14-1	double	40116600	601749000	15	7.4
SPIN30*	30-15-1	double	155117520	2481880320	16	30.4

	m_x-m_y-m_z	Communication distances ($n_{\text{nodes}} = 64$)	Message sizes [kB] ($n_{\text{nodes}} = 64$)
M1	1-1-n_{nodes}	$\pm1, -20, -19$	1050, 105, 0.128
M2	1-n_{nodes}-1	$\pm1, \pm20, \pm19$	1050, 48, 8
M3	n_{nodes}-1-1	±1	1050
M4	1-$\sqrt{n_{\text{nodes}}}$-$\sqrt{n_{\text{nodes}}}$	$\pm1, -20, \pm160, \pm159, -19$	1050, 105, 48, 8, 0.128
M5	$\sqrt{n_{\text{nodes}}}$-1-$\sqrt{n_{\text{nodes}}}$	$\pm1, -20, -19$	1050, 105, 0.128
M6	$\sqrt{n_{\text{nodes}}}$-$\sqrt{n_{\text{nodes}}}$-1	$\pm1, -20, -140, -19, -141$	1050, 48, 48, 8, 8
M7	2-4-$\sqrt{n_{\text{nodes}}}$	$\pm1, -20, -480, -160, -481, -159, -19$	1050, 105, 48, 48, 8, 8, 0.128
M8	2-$\sqrt{n_{\text{nodes}}}$-4	$\pm1, -20, -560, -80, -561, -79, -19$	1050, 105, 48, 48, 8, 8, 0.128
M9	$\sqrt{n_{\text{nodes}}}$-2-4	$\pm1, -20, -80, -79, -81, -19$	1050, 105, 96, 8, 8, 0.128

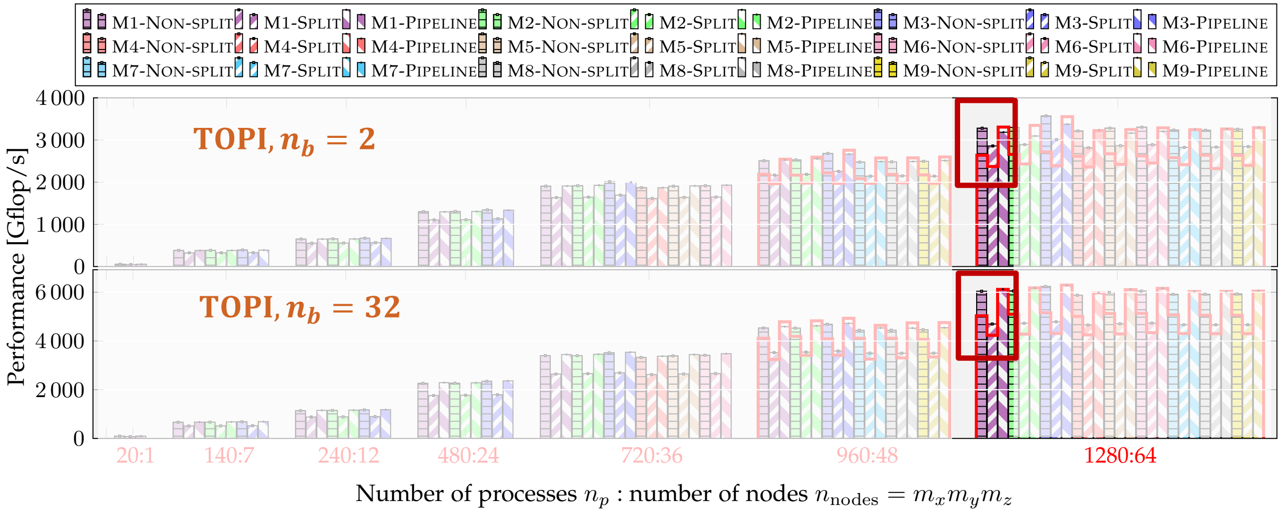
(a) TOPI-EHN

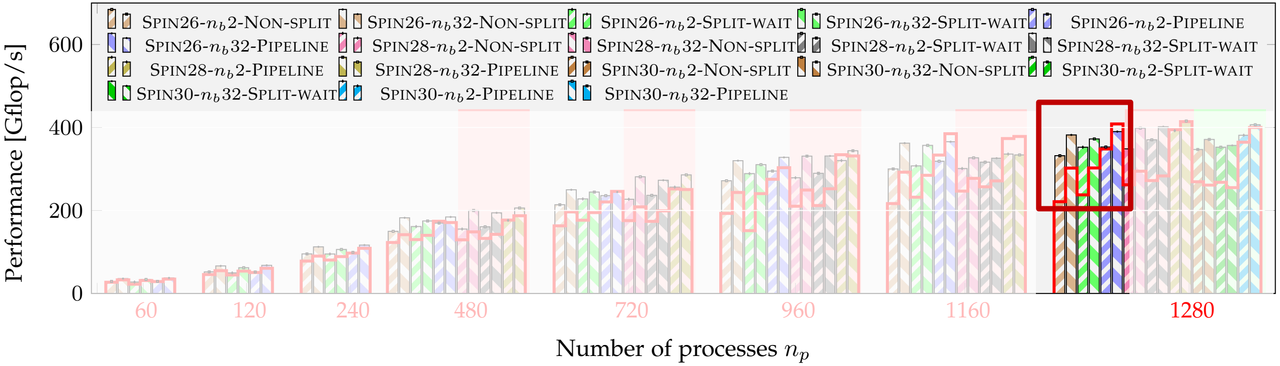


(b) SPIN26/30 (c) SPIN28



Performance: Hybrid parallel Chebychev Filter Diagonalization





Better overlap for decomposition of slow ide wave and large communication overhead

WITH BARRIER

Pipeline mode better (non-split suffer more for more saturating case)

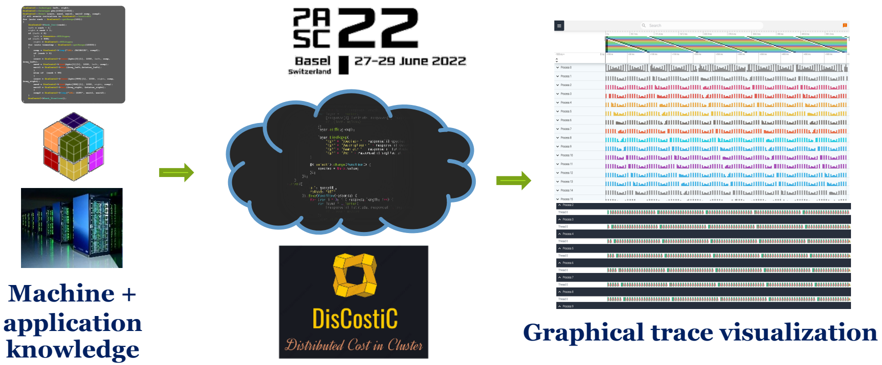
WITHOUT BARRIER

Non-split mode on-par with pipeline mode or even better for more saturating case

Overlapping via explicit programming techniques may not be necessary for strongly bandwidth-saturating code with large (but not dominant) communication overhead due to the presence of natural overlap by desynchronization

On-going and future work

SIMULATION FRAMEWORK

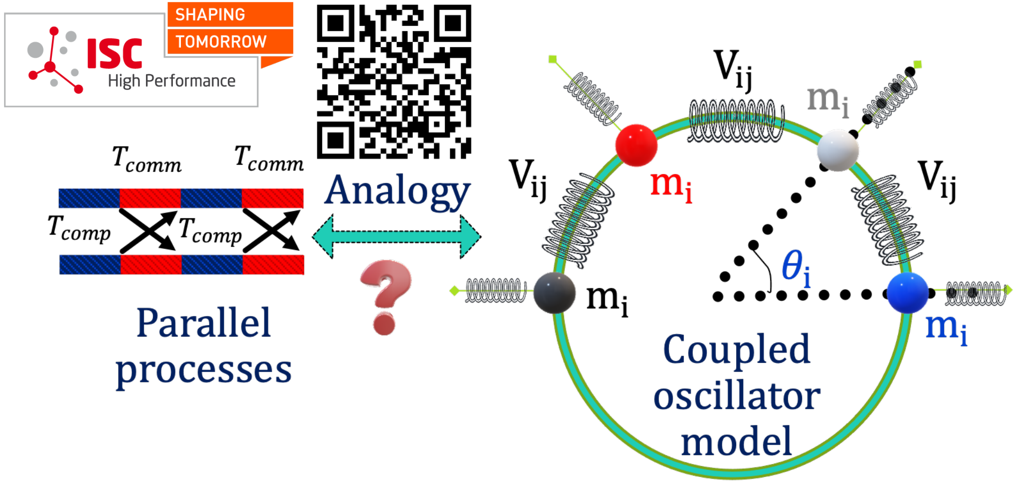


DisCostiC: A DSL-based Parallel Simulation Framework

Using First-Principles Analytic Performance Models

SIMULATION FRAMEWORK

PHYSICAL OSCILLATOR MODEL



Physical Oscillator Model For

Parallel Distributed Computing

On-going and future work

SIMULATION FRAMEWORK

PHYSICAL OSCILLATOR MODEL

MACHINE LEARNING TECHNIQUES AND ADVANCED METRICS FOR ANALYSIS

Machine Learning Techniques And Advanced Metrics For Analysis Of Parallel Programs

On-going and future work

SIMULATION FRAMEWORK

PHYSICAL OSCILLATOR MODEL

MACHINE LEARNING TECHNIQUES AND ADVANCED METRICS FOR ANALYSIS

THRESHOLD FOR LOAD BALANCING

Threshold Of Applying Scheduling Techniques For Load Balancing

University of Basel

Title: The Role of Idle Waves in Modeling and Optimization of Parallel Programs

Contact: Ayesha Afzal
ayesha.afzal@fau.de

Acknowledgement: Georg Hager, Gerhard Wellein

Thank you for Listening!

OMI4papps