Bo Kågström, Lennart Edblom, Lars Karlsson; Laura Grigori; Iain Duff, Jonathan Hogg; Jack Dongarra, and Nick Higham

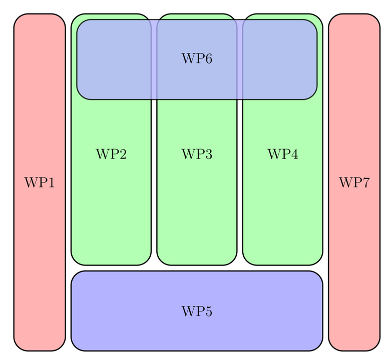
Umeå University, Sweden; Inria Paris-Rocqueancourt, France; RAL—Science Technology Facilities Council, UK; and University of Manchester, UK

NLAFET—Aim and Main Research Objectives

Aim: *Enable a radical improvement in the performance and scalability of a wide range of real-world applications relying on linear algebra software for future extreme-scale systems.*

- Development of novel *architecture-aware algorithms* that expose as much parallelism as possible, exploit heterogeneity, avoid communication bottlenecks, respond to escalating fault rates, and help meet emerging power constraints
- Exploration of *advanced scheduling strategies and runtime systems* focusing on the extreme scale and strong scalability in multi/many-core and hybrid environments
- Design and evaluation of novel strategies and software support for both *offline and online auto-tuning*
- Results will appear in the open source *NLAFET software library*

NLAFET Work Package Overview



- WP1: *Management and coordination*
- WP5: *Challenging applications—a selection*
  Materials science, power systems, study of energy solutions, and data analysis in astrophysics
- WP7: *Dissemination and community outreach*
  Research and validation results; stakeholder communities

WP2, WP3 and WP4 at a glance!

- Linear Systems Solvers
- Hybrid BLAS
- Eigenvalue Problem Solvers
- Singular Value Decomposition Algorithms

- Lower Bounds on Communication for Sparse Matrices
- Direct Methods for (Near–)Symmetric Systems
- Direct Methods for Highly Unsymmetric Systems
- Hybrid Direct–Iterative Methods

- Computational Kernels for Preconditioned Iterative Methods
- Iterative Methods: use p vectors per it, nearest-neighbor comm
- Preconditioners: multi-level, comm. reducing

Research Focus—Critical set of fundamental LA operations

- WP2: *Dense linear systems and eigenvalue problem solvers*
- WP3: *Direct solution of sparse linear systems*
- WP4: *Communication-optimal algorithms for iterative methods*
- WP6: *Cross-cutting issues*

WP2, WP3 and WP4: research into extreme-scale parallel algorithms
WP6: research into methods for solving common cross-cutting issues

WP6: Cross-cutting issues and challenges!

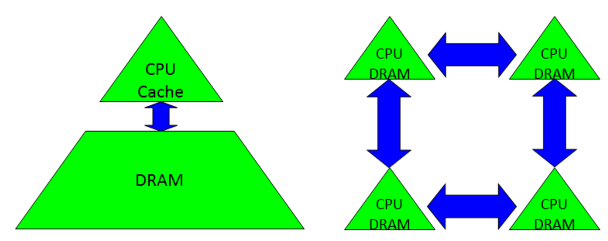
Extreme-scale systems are hierarchical and heterogeneous in nature!

- Scheduling and Runtime Systems:
  - Task-graph-based multi-level scheduler for multi-level parallelism
  - Investigate user-guided schedulers: application-dependent balance between locality, concurrency, and scheduling overhead
  - Run-time system based on parallelizing critical tasks ($Ax = \lambda Bx$)
  - Address the thread-to-core mapping problem
- Auto-Tuning:
  - Off-line: tuning of critical numerical kernels across hybrid systems
  - Run-time: use feedback during and/or between executions on similar problems to tune in later stages of the algorithm
- Algorithm-Based Fault Tolerance:
  - Explore new NLA methods of resilence and develop algorithms with these capabilities.

Avoid Communications—extreme-scale systems accentuate the need!

Algorithms have two costs (measured in time or energy):

1. Arithmetic (Flops)
2. Communication: moving data between
   - levels of a memory hierarchy (sequential case)
   - processors over a network (parallel case).



Running time of an algorithm involves three terms:

- # Flops ∗ Time_per_flop
- # Words moved / Bandwidth
- # Messages ∗ Latency

Time_per_flop $\lll$ **1**/ Bandwidth $\lll$ Latency

Gaps growing "exponential" with time

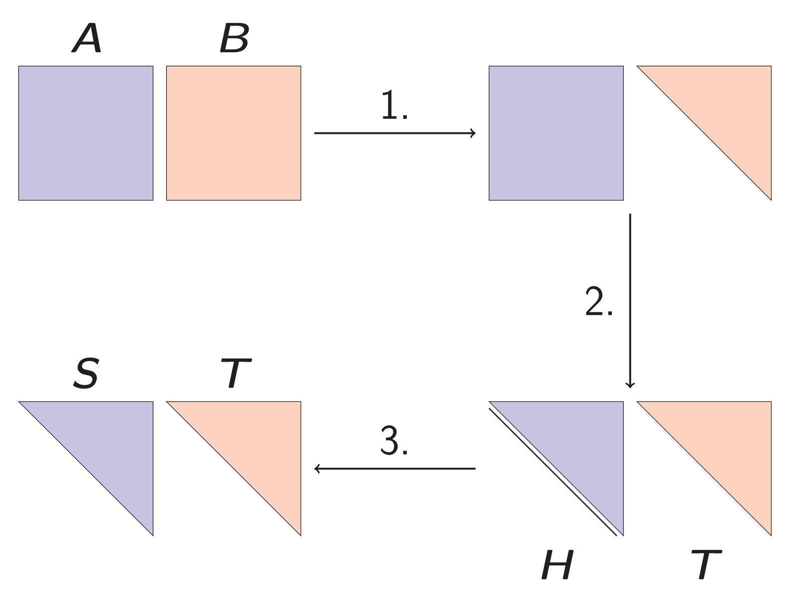
Annual improvements

Time per flop		*Bandwidth*	*Latency*
59%	Network	26%	15%
	DRAM	23%	5%

Goal: Redesign algorithms (or invent new) to avoid communication!
Attain lower bounds on communication if possible!

Generalized eigenvalue problem—need for autotuning!

Find pairs of eigenvalues λ and eigenvectors x s.t.
$Ax = \lambda Bx$



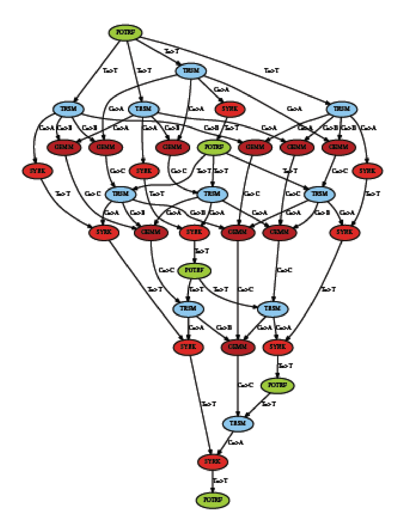
1. QR factorization
2. Hessenberg-Triangular reduction
3. QZ algorithm (generalized Schur decomposition)

Tunable parameters in state-of-the-art parallel QZ algorithm:

n_{min1}	algorithm selection threshold
n_{min2}	algorithm selection threshold
n_{min3}	parallelization threshold
P_{AED}	# processors for subproblems
`MMULT`	level-3 BLAS threshold
`NCB`	cache-blocking block size
`NIBBLE`	loop break threshold
n_{AED}	AED deflation window size
n_{shift}	#shifts per iteration
`NUMWIN`	# deflation windows
`WINEIG`	eigenvalues per window
`WINSIZE`	window size
`WNEICR`	#eigenvalues moved together

Task-graph-based scheduling and run-time systems

- Express algorithmic dataflow, *not* explicit data movement
- Blocked Cholesky tasks: POTRF, TRSM, GEMM, SYRK
- PTG representation: symbolic, problem size independent



- Data flow based execution using PaRSEC (ICL-UTK)
- Assigns computations threads to cores; overlaps comm. & comp.
- Distributed dynamic scheduler based on NUMA nodes and data reuse

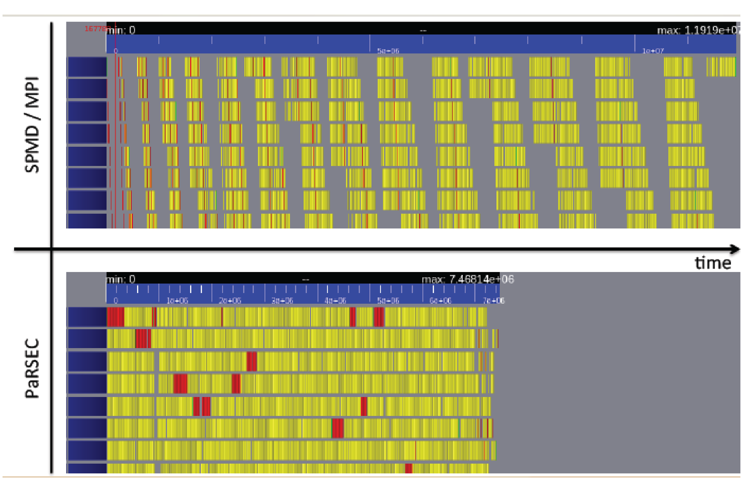


Figure 1: Cholesky PTG run by PaRSEC; 45% improvement

Acknowledgments

- NLAFET has received funding from the European Unions Horizon 2020 Research and Innovation Programme under Grant agreement No 671633.

Contact Information

- NLAFET Coordinator: Prof. Bo Kågström (bokg@cs.umu.se)
  Dept. of Computing Science and HPC2N, Umeå University, Sweden
- Web-site: http://www.nlafet.eu Email: info@nlafet.eu

NLAFET Partners