Myths and Legends of Ancient Parallel Computing: Load Balancing

Alexey Lastovetsky

University College Dublin, Ireland

PPAM 2017
September 11, 2017

Outline

Load balancing: Intuition

Load balancing is a performance optimization technique widely used in parallel computing.

What is the intuition behind load balancing?

If the load of processors is balanced:

- Processors do not wait at points of data exchange and synchronization
  - => They never waste processor cycles on waiting
  - => The always do useful work
  - => Parallel computation time is minimal

Load balancing: Analysis

Let us formulate the intuition mathematically (simplest case):

- p identical processors
- $s(x) = \frac{x}{t(x)}$, where $t(x)$ is the execution time of workload x
- If $\forall \Delta x > 0 : \frac{s(x)}{x} \geq \frac{s(x+\Delta x)}{x+\Delta x}$, then balancing the load of the processors will minimize the parallel computation time*

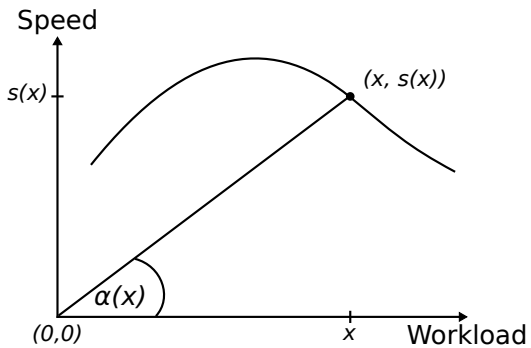


Figure: Speed function suitable for optimization through load balancing. $\alpha(x)$ is reversely proportional to computation time.

* A. Lastovetsky, L. Szustak, and R. Wyrzykowski, "Model-based optimization of EULAG kernel on Intel Xeon Phi through load imbalancing", IEEE Transactions on Parallel and Distributed Systems 28(3):787-797, 2017.

Load balancing: Evidence

Do we have evidence for performance profiles being that nice?
Actually, yes:

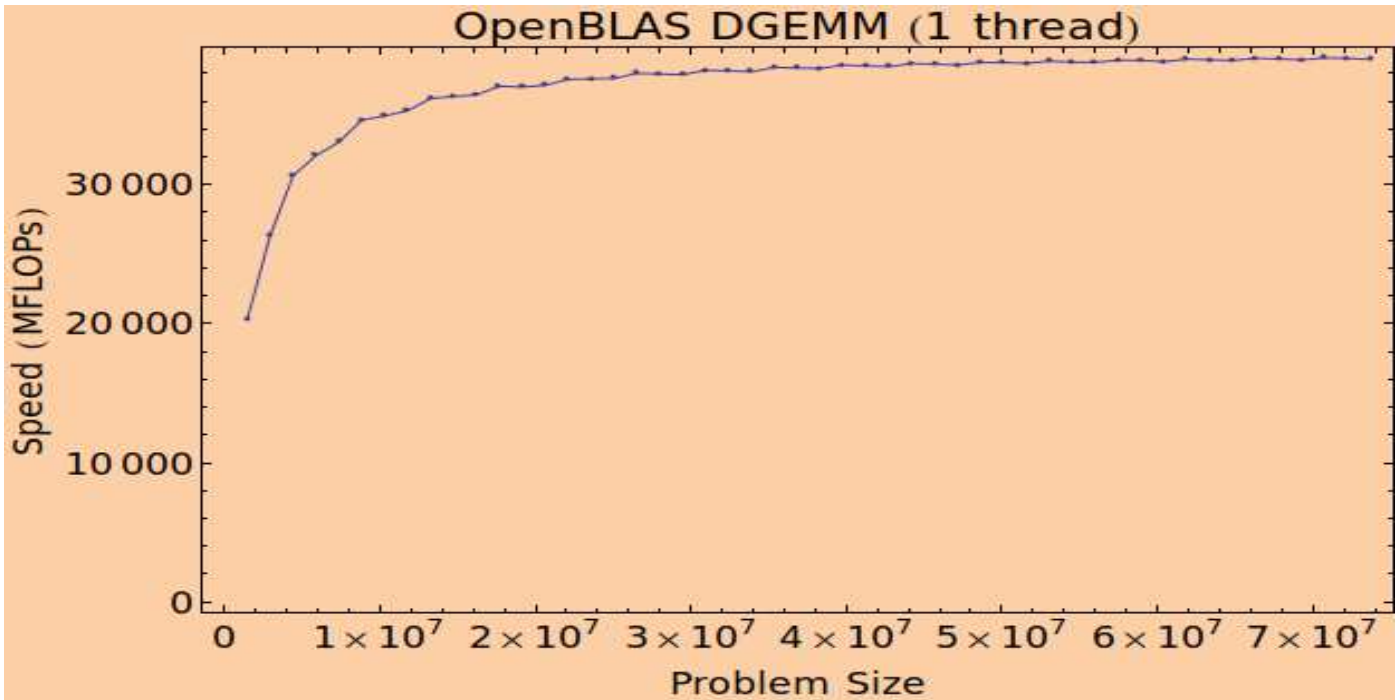


Figure: Speed function of OpenBLAS DGEMM application executed on a single core on the Intel Haswell server.

Load balancing: Counter-evidence

Same application but in the muticore era (MCE):

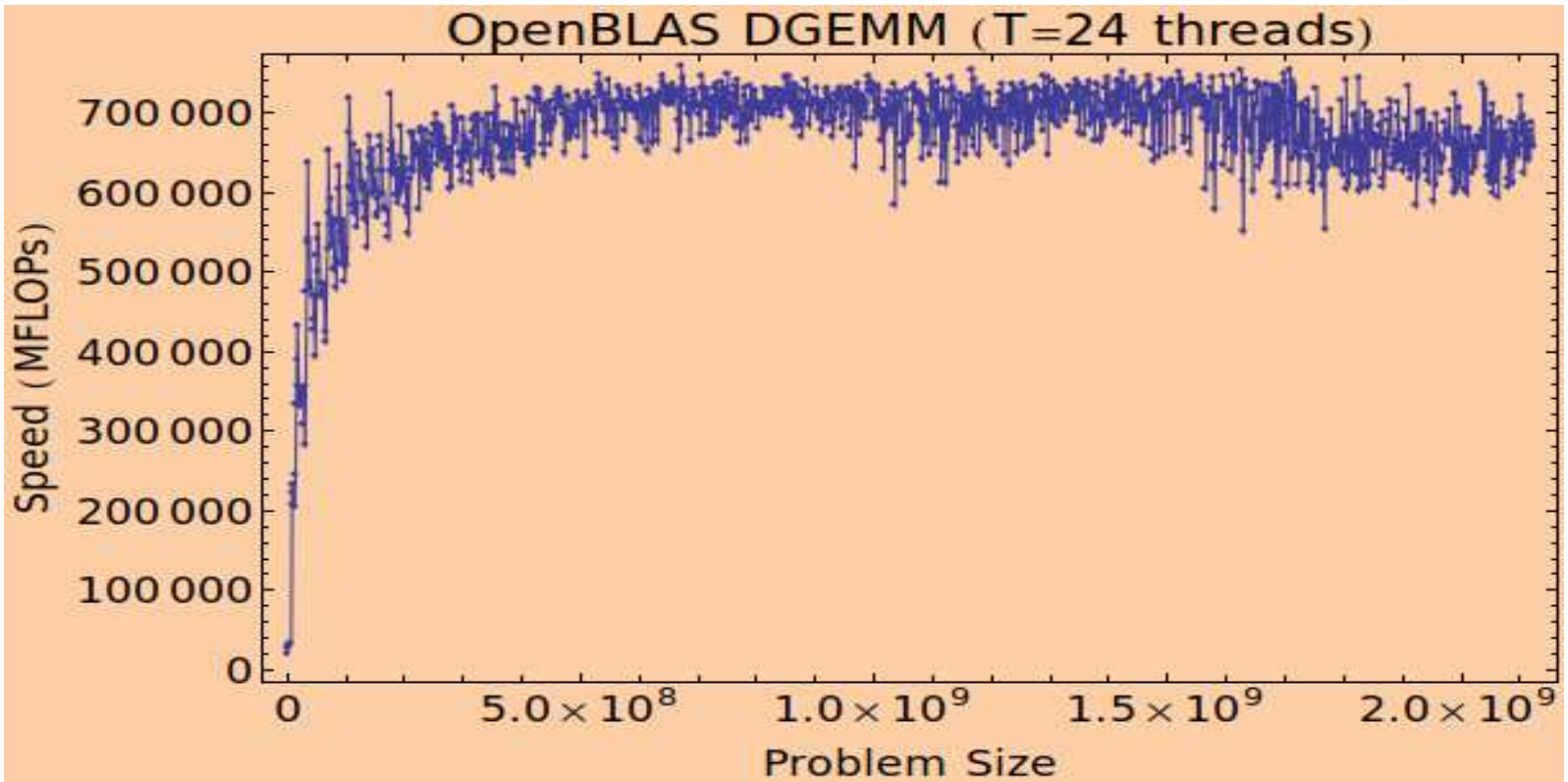


Figure: OpenBLAS DGEMM executing 24 threads on 24-core CPU of the Intel Haswell server.

Load balancing: More counter-evidence

FFTW application computing $2D$ discrete Fourier transform (DFT) of size $n \times n$:

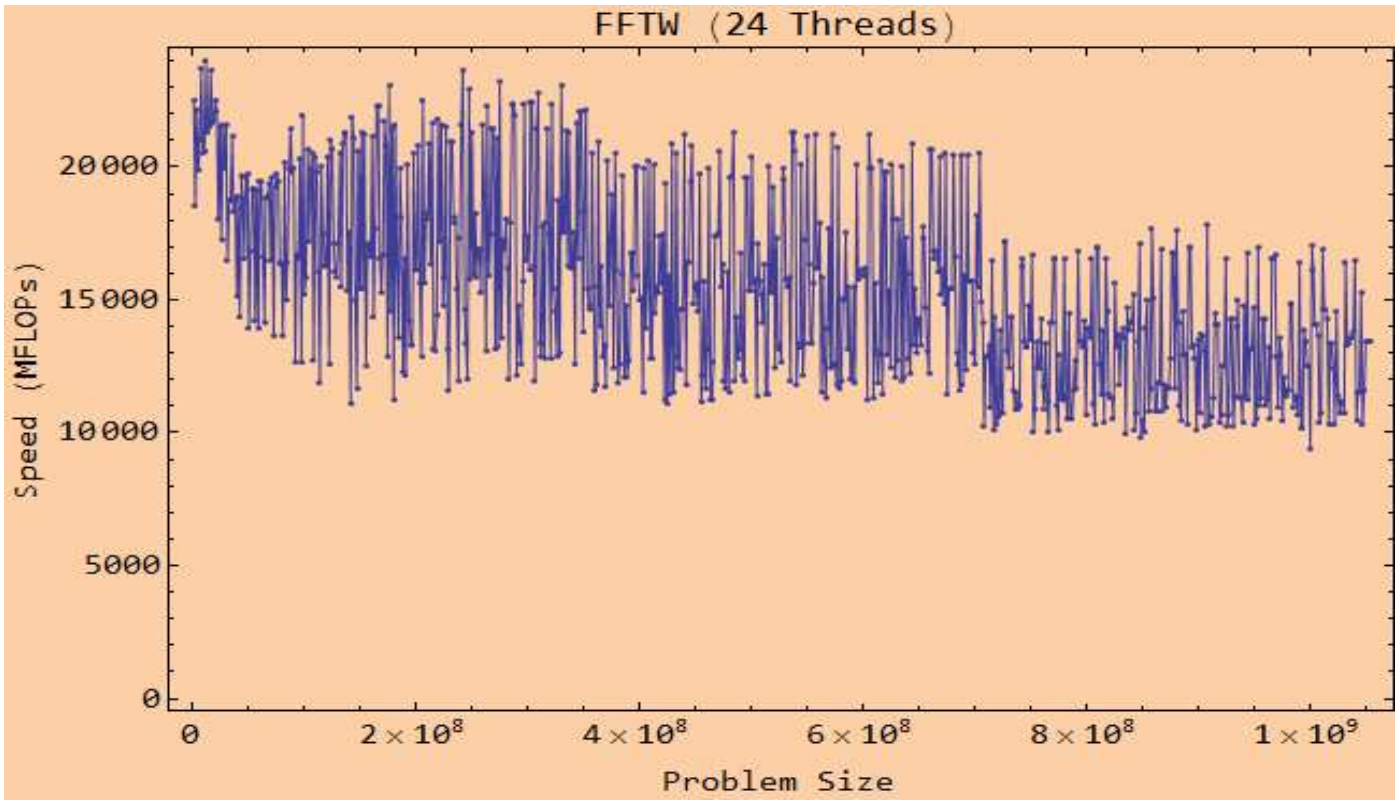


Figure: FFTW executing 24 threads on 24-core CPU of the server.

Load (im)balancing: Analysis

More threads/cores => bigger variations:

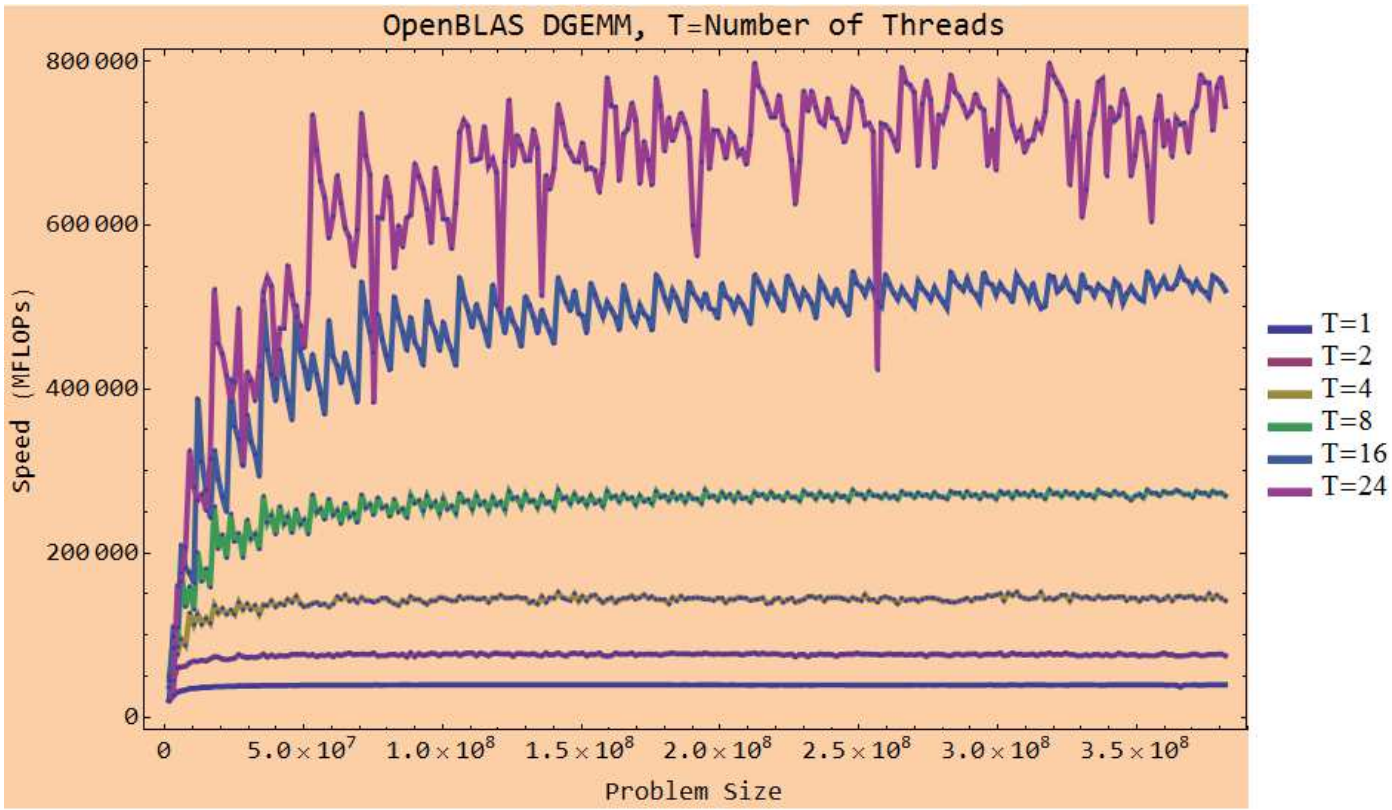


Figure: Speed function of OpenBLAS DGEMM application executing varying number of threads (T) on the Intel Haswell server.

Load (im)balancing: Analysis

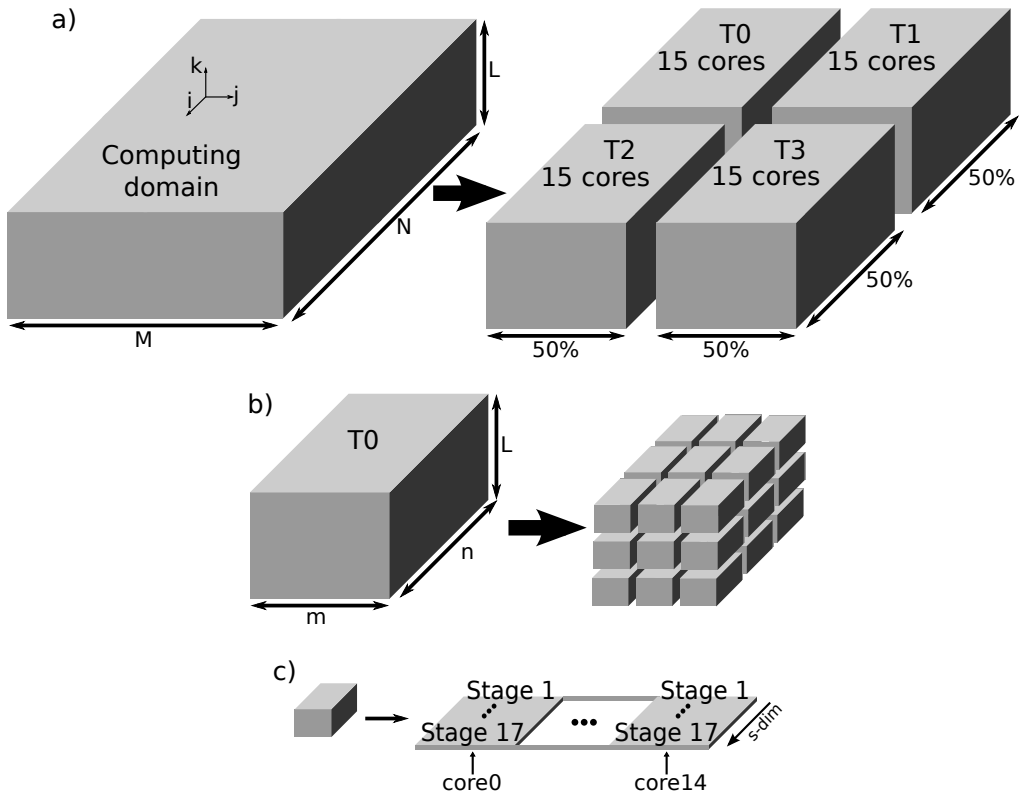
Summary of observations and results*:

- Widths of speed variations can be significant (with averages around 17% DGEMM and 60% for FFTW)
- The variations do not decrease with the increase of problem size
- Efficient workload distribution algorithm, POPTA, has been proposed that returns an optimal distribution that minimizes the parallel execution time but not necessarily balances the load of the processors
- The average and maximum performance improvements of POPTA solutions over the even (load-balanced) distribution are $(13\%, 71\%)$ for DGEMM and $(40\%, 95\%)$ for FFTW

* A. Lastovetsky and R. Reddy, "New Model-based Methods and Algorithms for Performance and Energy Optimization of Data Parallel Applications on Homogeneous Multicore Clusters", IEEE Transactions on Parallel and Distributed Systems 28(4):1119-1133, 2017.

3D MPDATA on Intel Xeon Phi

MPDATA is a CFD solver performing stencil computations.
Parallel MPDATA for Xeon Phi uses hierarchical data decomposition*



* L. Szustak, et al. "Adaptation of MPDATA heterogeneous stencil computation to Intel Xeon Phi coprocessor", Scientific Programming 2015, Article 10.

3D MPDATA on Intel Xeon Phi

Speed function of team T_0 built in parallel with other teams*:

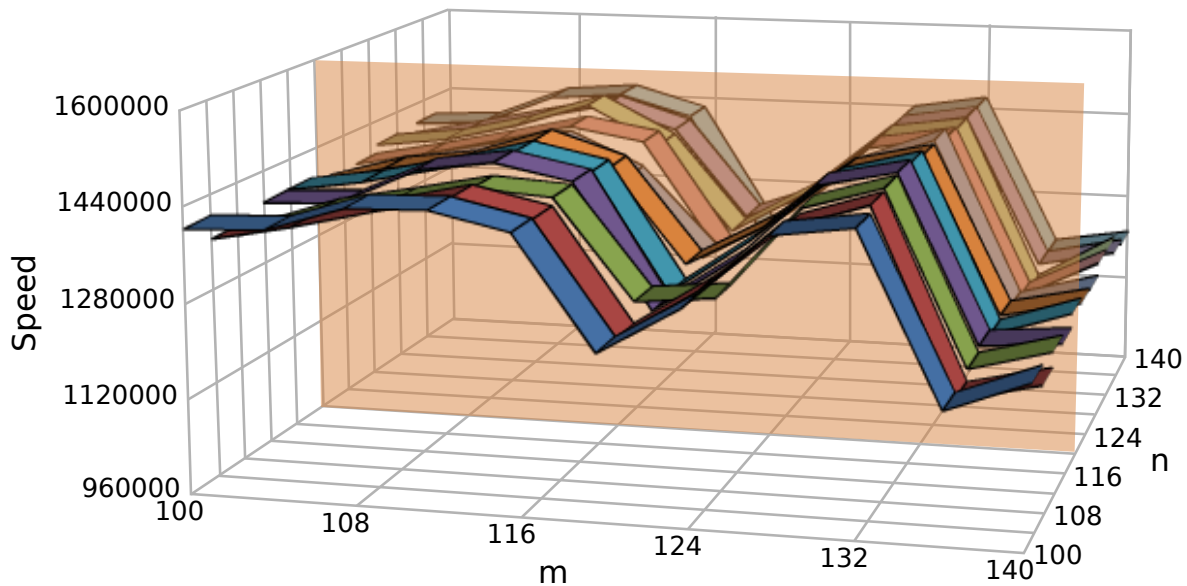
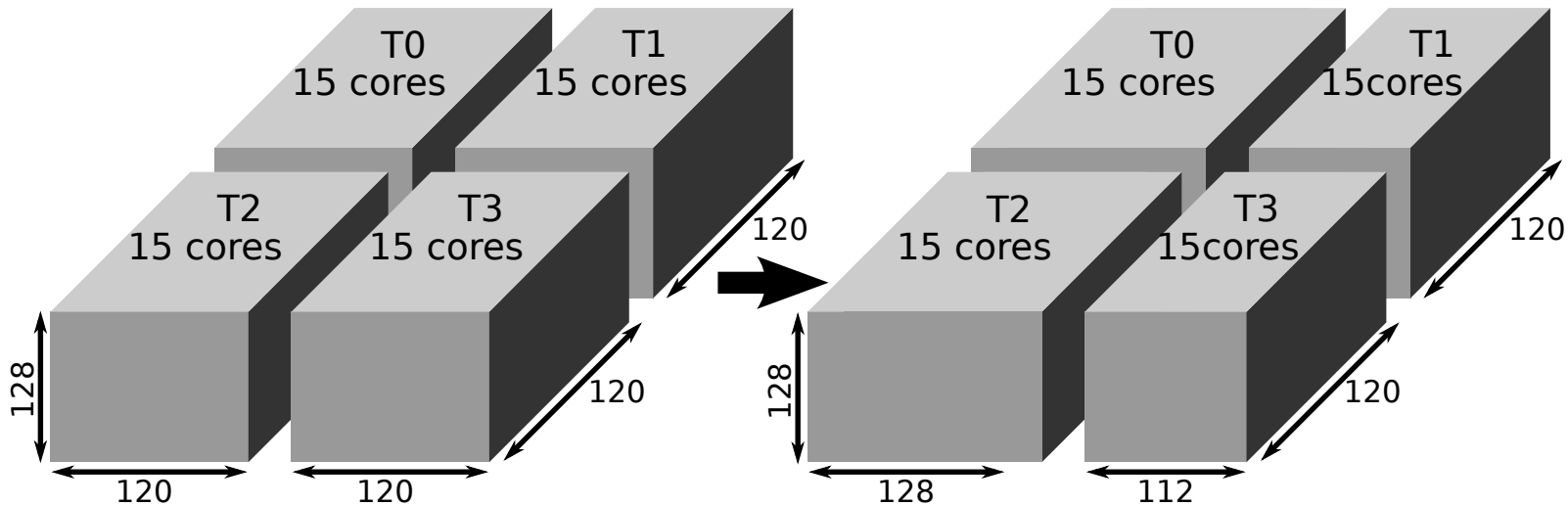


Figure: Speed of execution of MPDATA workload by team T_0 as function of n and m ($l = 128$). Very little dependence on n.

* A. Lastovetsky, L. Szustak, and R. Wyrzykowski, "Model-based optimization of EULAG kernel on Intel Xeon Phi through load imbalancing", IEEE Transactions on Parallel and Distributed Systems 28(3):787-797, 2017.

3D MPDATA on Intel Xeon Phi

The idea is to use *n* from the load balanced solution and find new values of *m* for even and odd teams minimizing the execution time*



* A. Lastovetsky, L. Szustak, and R. Wyrzykowski, "Model-based optimization of EULAG kernel on Intel Xeon Phi through load imbalancing", IEEE Transactions on Parallel and Distributed Systems 28(3):787-797, 2017.

3D MPDATA on Intel Xeon Phi

Speed functions obtained by cutting speed surfaces for T_0, T_1, T_2, and T_3 by plane $n = 120$*:

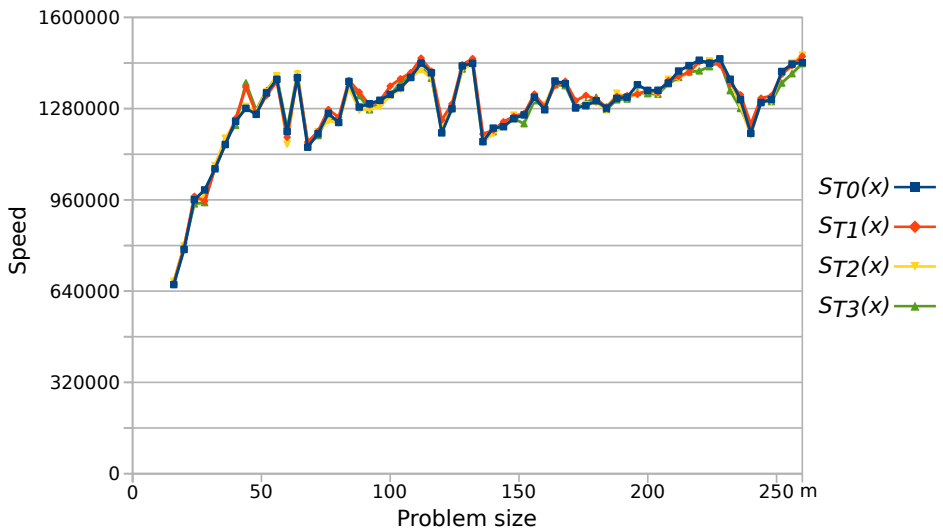
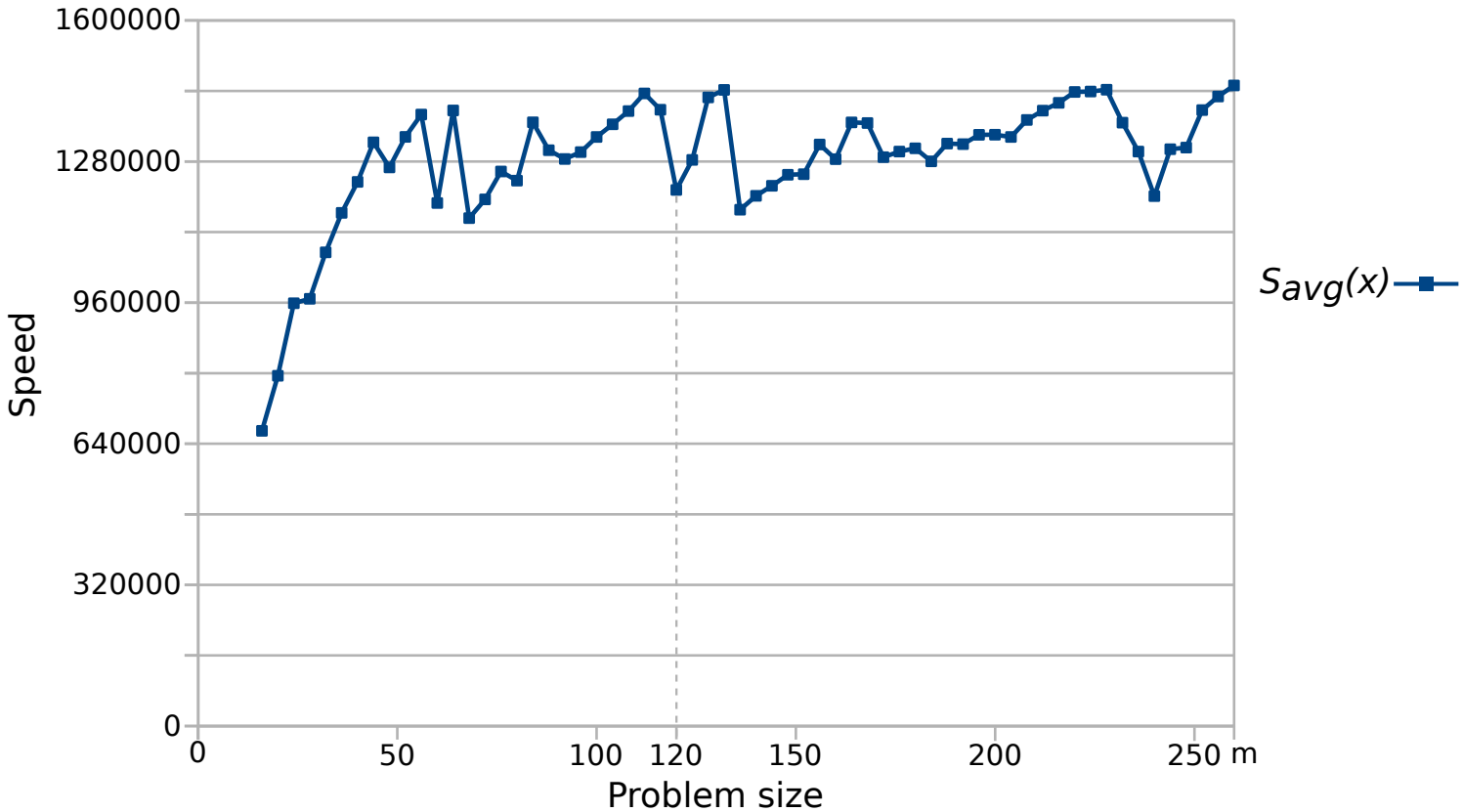


Figure: Speeds of four teams built simultaneously as functions of parameter m ($n = 120$ and $l = 128$)

* A. Lastovetsky, L. Szustak, and R. Wyrzykowski, "Model-based optimization of EULAG kernel on Intel Xeon Phi through load imbalancing", IEEE Transactions on Parallel and Distributed Systems 28(3):787-797, 2017.

3D MPDATA on Intel Xeon Phi

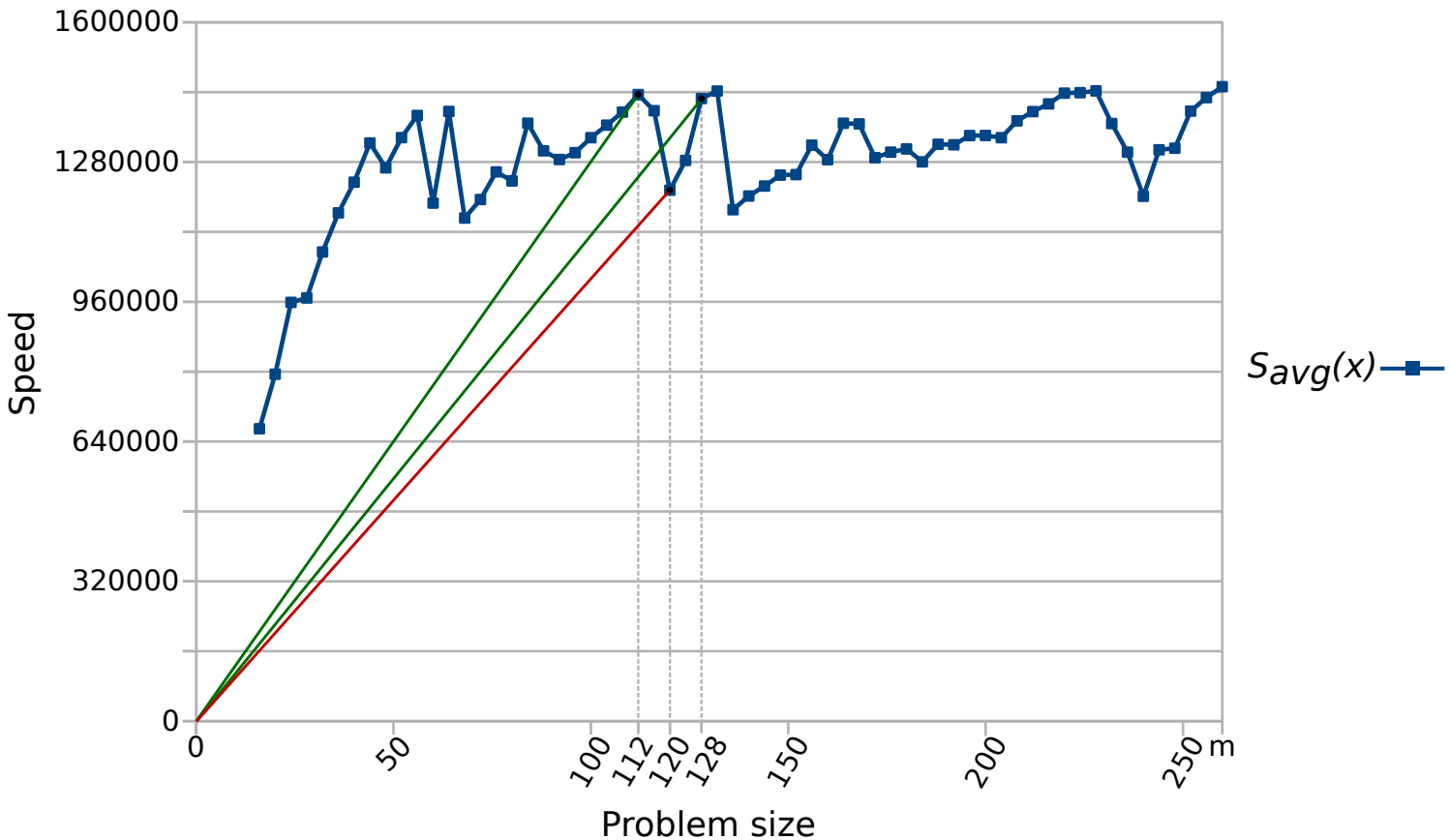
Average speed function*:



* A. Lastovetsky, L. Szustak, and R. Wyrzykowski, "Model-based optimization of EULAG kernel on Intel Xeon Phi through load imbalancing", IEEE Transactions on Parallel and Distributed Systems 28(3):787-797, 2017.

3D MPDATA on Intel Xeon Phi

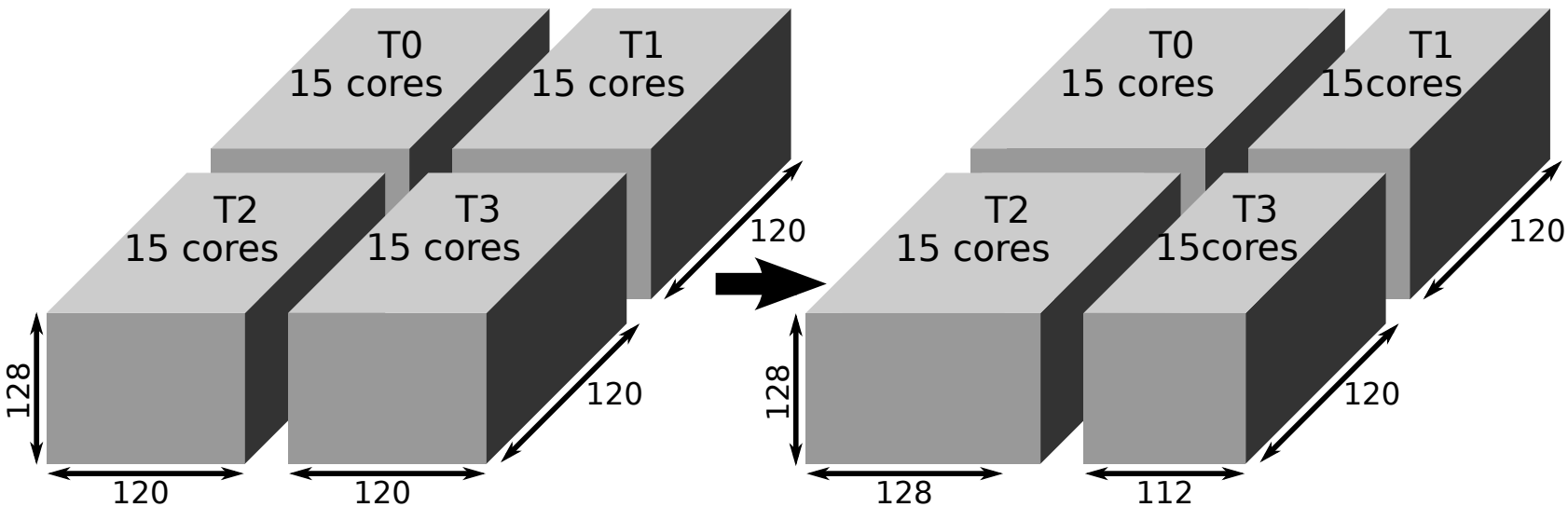
Optimal solution*:



* A. Lastovetsky, L. Szustak, and R. Wyrzykowski, "Model-based optimization of EULAG kernel on Intel Xeon Phi through load imbalancing", IEEE Transactions on Parallel and Distributed Systems 28(3):787-797, 2017.

3D MPDATA on Intel Xeon Phi

Optimal solution*:



* A. Lastovetsky, L. Szustak, and R. Wyrzykowski, "Model-based optimization of EULAG kernel on Intel Xeon Phi through load imbalancing", IEEE Transactions on Parallel and Distributed Systems 28(3):787-797, 2017.

3D MPDATA on Intel Xeon Phi

Pitfalls: the impact of resource sharing

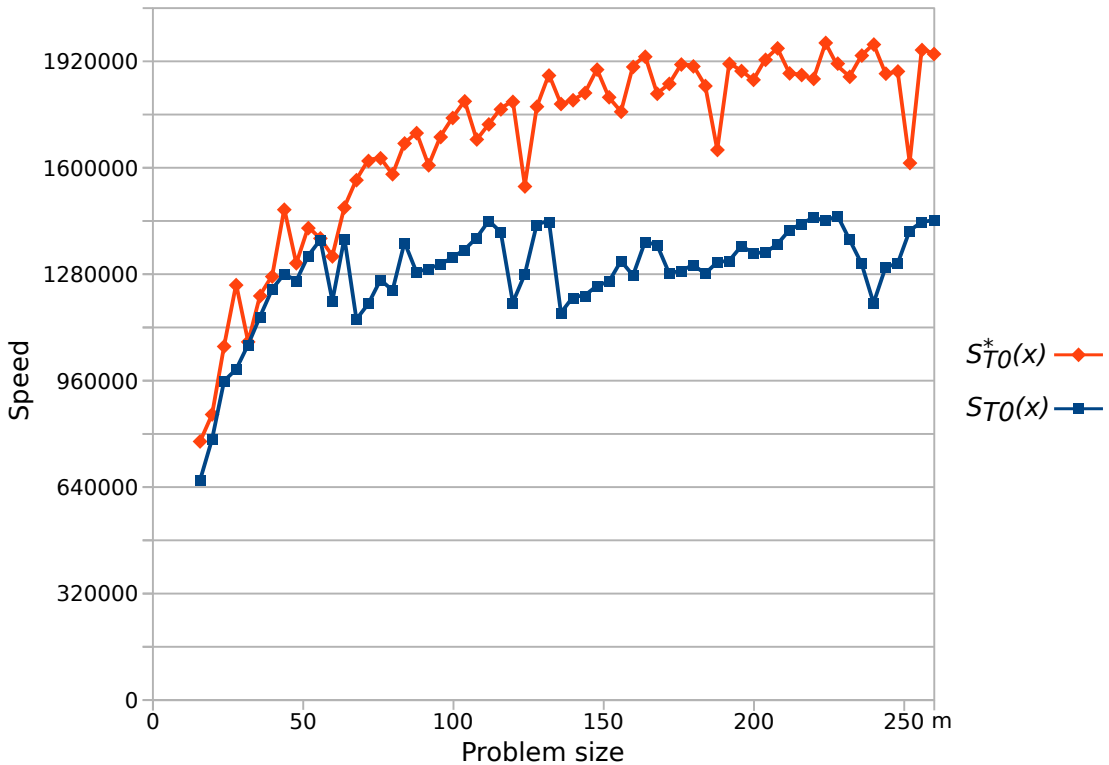


Figure: Comparison of speed functions of team T_0, measured separately ($S^*_{T_0}(x)$) and simultaneously with other three teams ($S_{T_0}(x)$) executing the same workload

3D MPDATA on Intel Xeon Phi: Experimental results

Table: Times for different decompositions of MPDATA $240 \times 240 \times 128$ domain: $120 \times (120 + \Delta m) \times 128$ sub-domains are processed by odd teams, while $120 \times (120 - \Delta m) \times 128$ sub-domains are processed by even teams

Δm	Theoretical time [s]	Experimental time [s]	Speedup
0	1.486	1.548	1.000
4	1.470	1.470	1.053
6	1.401	1.374	1.127
7	1.422	1.361	1.137
8	1.386	1.364	1.135
9	1.398	1.348	1.148
10	1.397	1.352	1.145
11	1.429	1.372	1.129
12	1.402	1.368	1.131

3D MPDATA on Intel Xeon Phi: Experimental results

Table: Experimental time for all teams with different partitionings: the odd teams process sub-domains of size $120 \times (120 + \Delta m) \times 128$, while the even teams process sub-domains of size $120 \times (120 - \Delta m) \times 128$.

Offset Δm	Experimental time [s]				
	Team 0	Team 1	Team 2	Team 3	Total
0	1.515	1.498	1.518	1.503	1.548
4	1.456	1.247	1.455	1.249	1.470
6	1.364	1.161	1.359	1.162	1.374
7	1.355	1.161	1.341	1.168	1.361
8	1.355	1.166	1.349	1.172	1.364
9	1.340	1.155	1.335	1.161	1.348
10	1.345	1.141	1.337	1.152	1.352
11	1.363	1.156	1.357	1.154	1.372
12	1.360	1.163	1.353	1.165	1.368

3D MPDATA on Intel Xeon Phi

Self-adaptable implementation of MPDATA*:

- MPDATA is used for long running simulations (thousands time steps)
- Execution speed is stable – speed function of one time step is the same for any time step
- First few time steps can be used not only for calculations but also for building the speed function for a limited range around the balanced solution
- Then the built speed function is used as input to POPTA
- Negligible overhead (requires <20 time steps and <2% of the execution time of one time step)
  - In numerical 2-day weather prediction (over 16000 time steps), the overhead will be less than 0.005%

* A. Lastovetsky, L. Szustak, and R. Wyrzykowski, "Model-based optimization of EULAG kernel on Intel Xeon Phi through load imbalancing", IEEE Transactions on Parallel and Distributed Systems 28(3):787-797, 2017.

Energy consumption in single core era

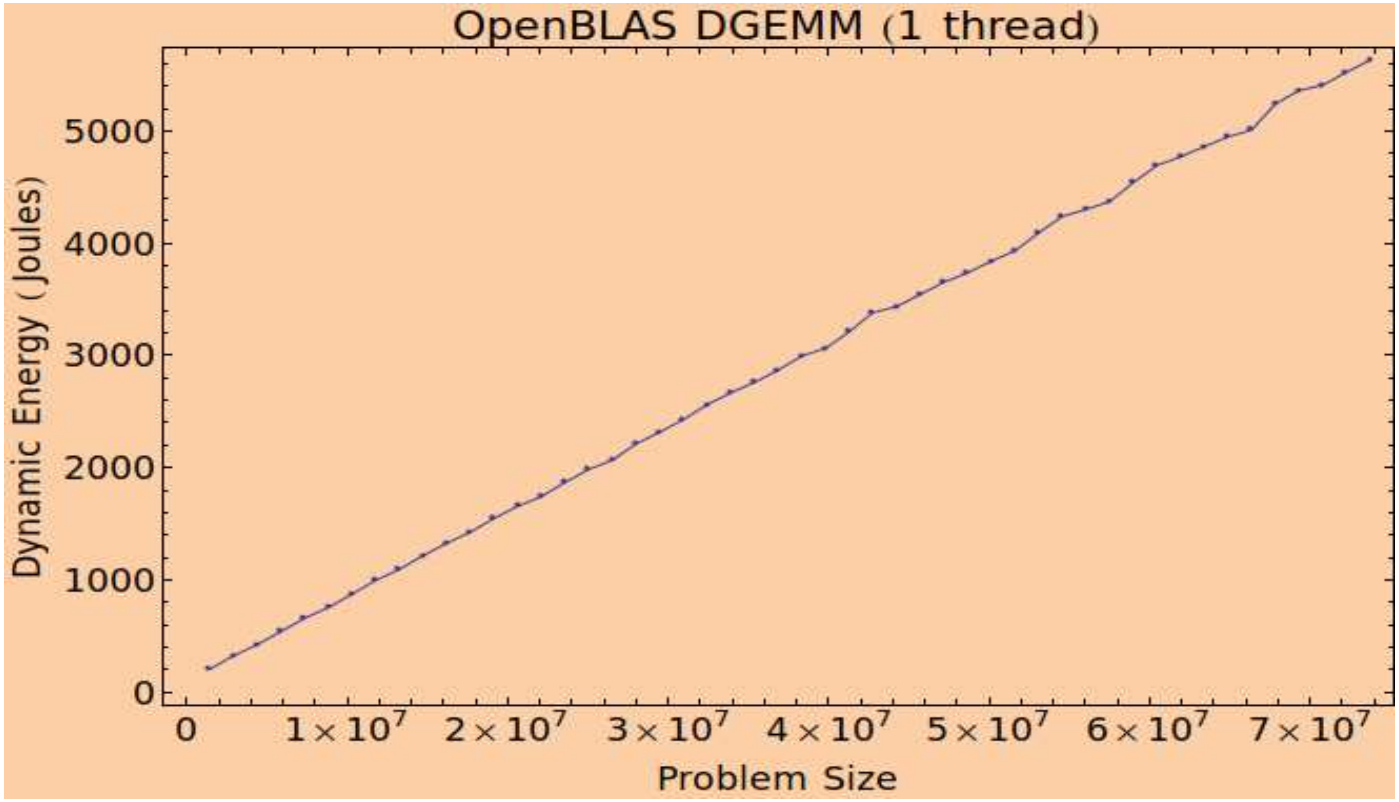


Figure: Dynamic energy consumption of OpenBLAS DGEMM application executed on a single core on the Intel Haswell server.

Energy consumption in multicore era

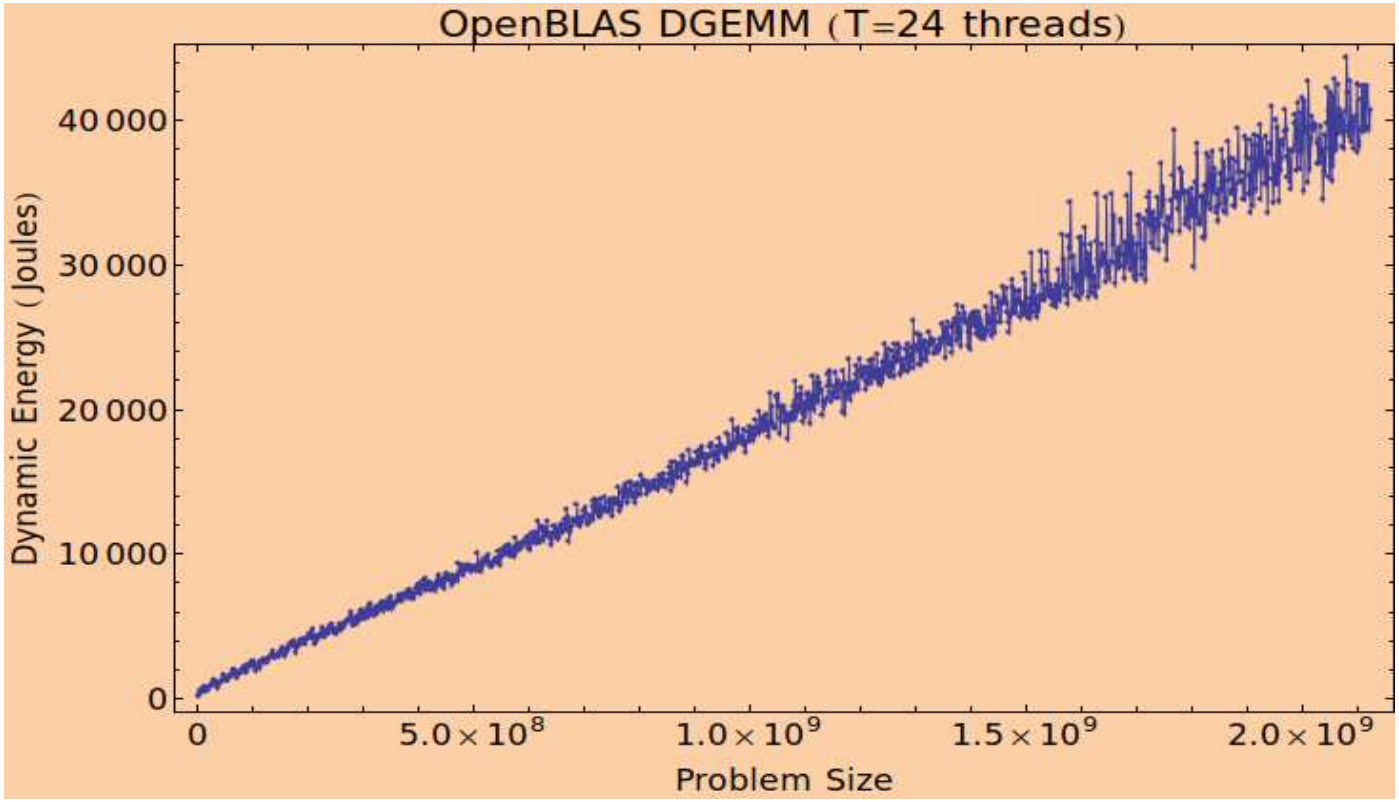


Figure: Function of dynamic energy consumption against problem size for OpenBLAS DGEMM application for $T = 24$ on the Intel Haswell server.

Energy consumption in multicore era

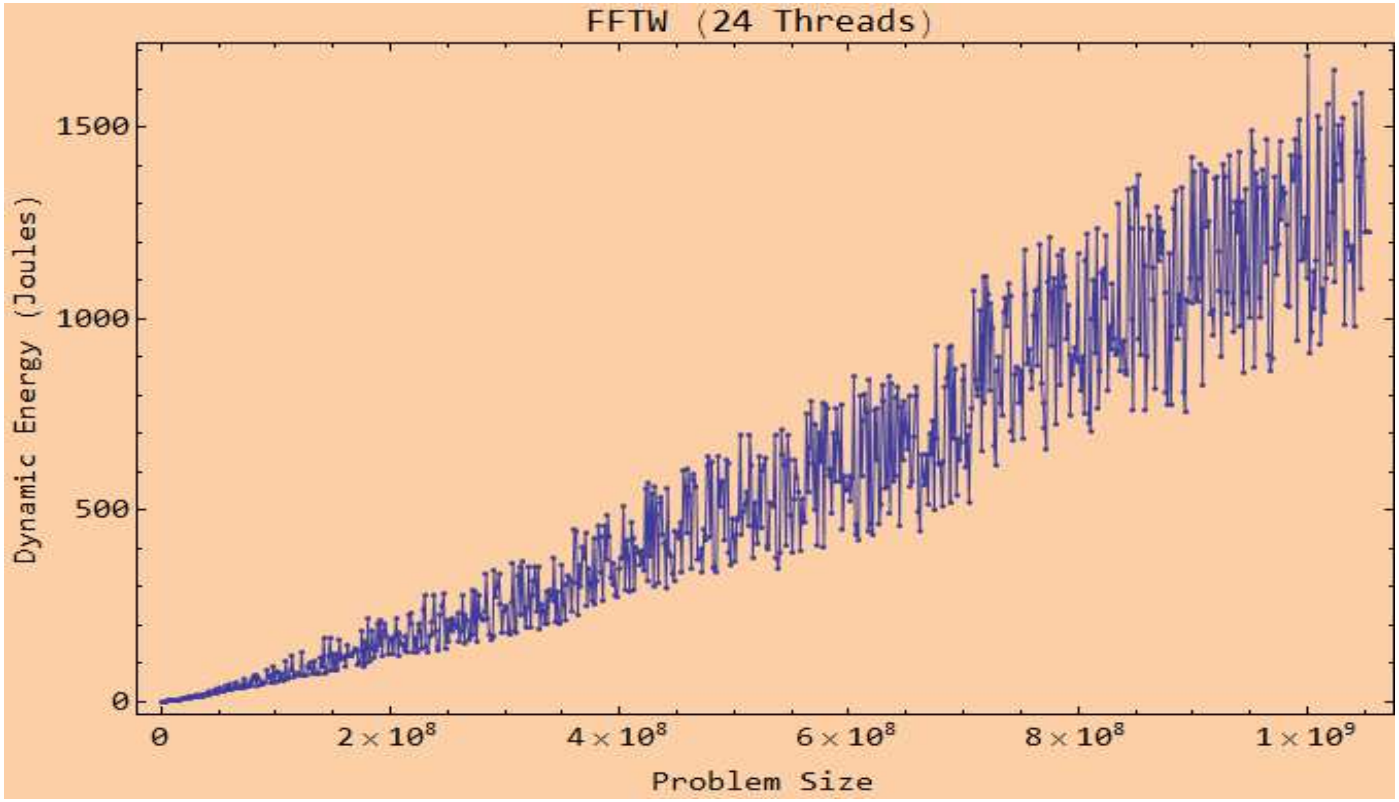


Figure: Function of dynamic energy consumption against problem size for FFTW executing 24 threads on the Intel Haswell server.

Optimization for energy in multicore era

Summary of observations and results*:

- Widths of variations in dynamic energy consumption can be very significant (up to 70% for DGEMM and 125% for FFTW)
- The variations increase with the increase of problem size
- Efficient workload distribution algorithm, EOPTA, has been proposed that returns an optimal distribution that minimizes the energy consumption but not necessarily balances the load of the processors
- The average and maximum reductions in dynamic energy consumption of EOPTA solutions over the even (load-balanced) distribution are $(18\%, 71\%)$ for DGEMM and $(22\%, 127\%)$ for FFTW

* A. Lastovetsky and R. Reddy, "New Model-based Methods and Algorithms for Performance and Energy Optimization of Data Parallel Applications on Homogeneous Multicore Clusters", IEEE Transactions on Parallel and Distributed Systems 28(4):1119-1133, 2017.

Optimization for performance and energy

Observations on single-objective optimizations through load distribution*:

- Optimization for performance only also reduces the energy consumption: $(12\%, 68\%)$ for DGEMM and $(22\%, 55\%)$ for FFTW
- Optimization for energy only significantly degrades the performance: by 95-100% for both DGEMM and FFTW

* A. Lastovetsky and R. Reddy, "New Model-based Methods and Algorithms for Performance and Energy Optimization of Data Parallel Applications on Homogeneous Multicore Clusters", IEEE Transactions on Parallel and Distributed Systems 28(4):1119-1133, 2017.

Bi-objective optimization for performance and energy

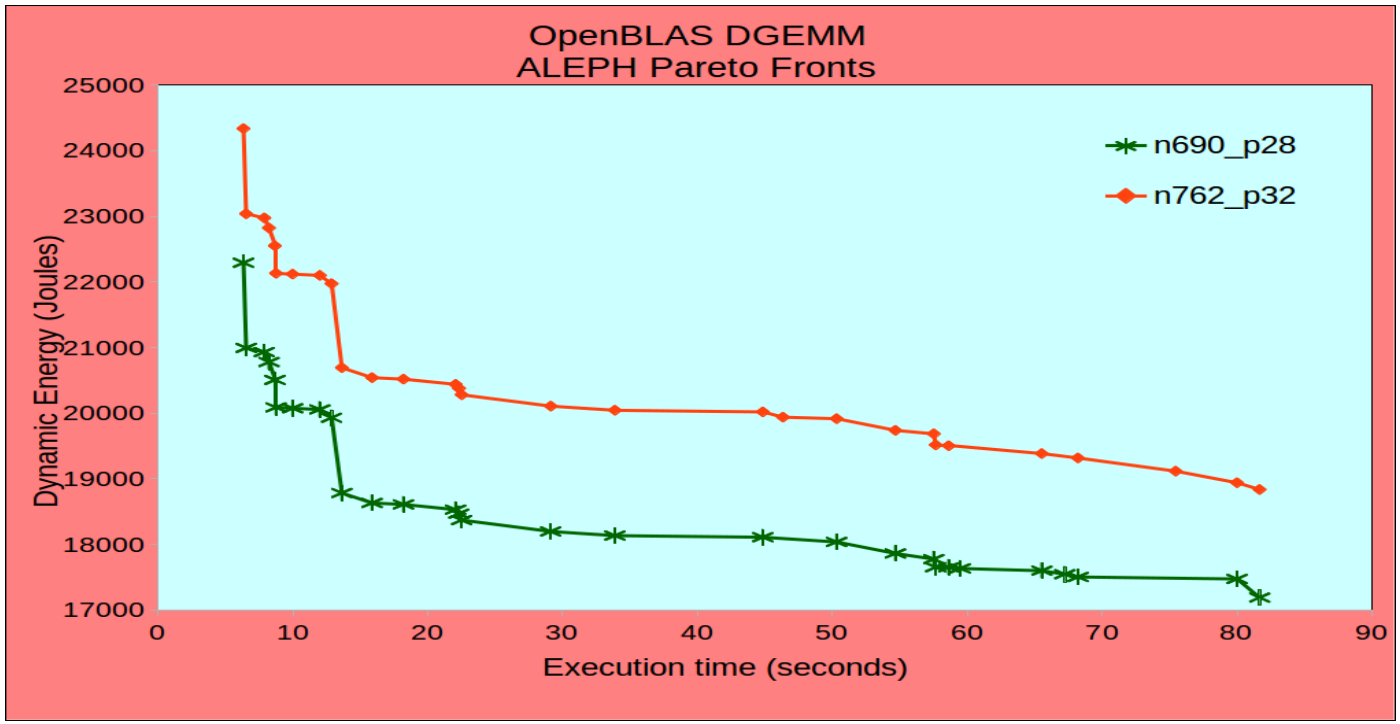


Figure: Globally Pareto-optimal set of solutions determined by *ALEPH* for OpenBLAS DGEMM.

* R. Reddy and A. Lastovetsky, "Bi-Objective Optimization of Data-Parallel Applications on Homogeneous Multicore Clusters for Performance and Energy", IEEE Transactions on Computers, vol. PP, issue 99: IEEE, 08/2017.

Bi-objective optimization for performance and energy

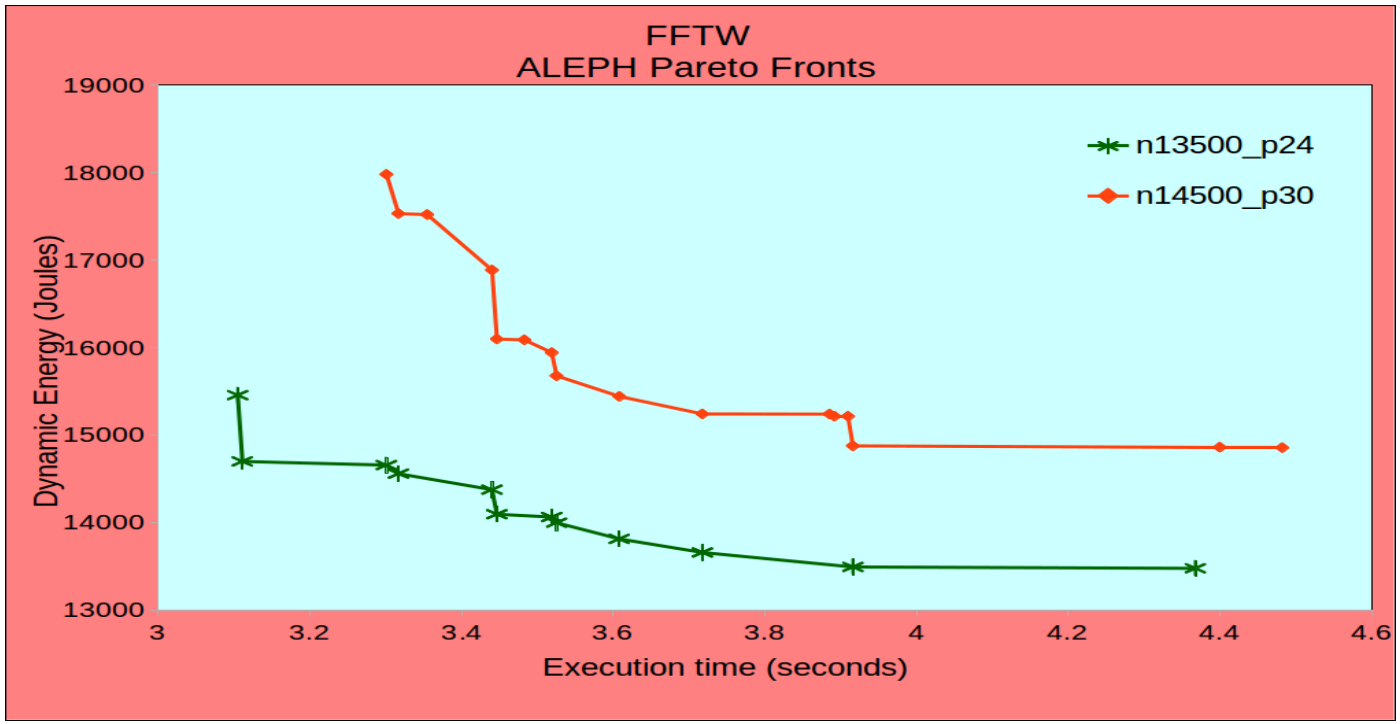
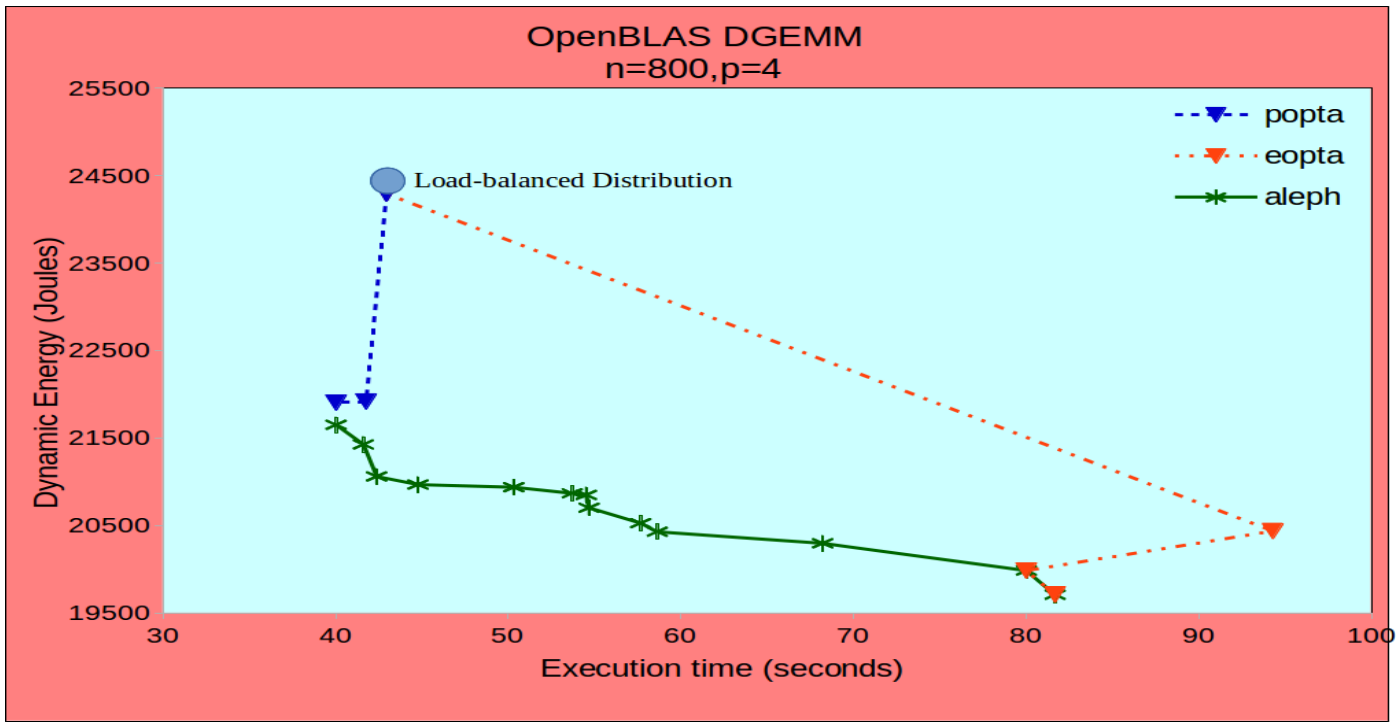


Figure: Globally Pareto-optimal set of solutions determined by *ALEPH* for FFTW application.

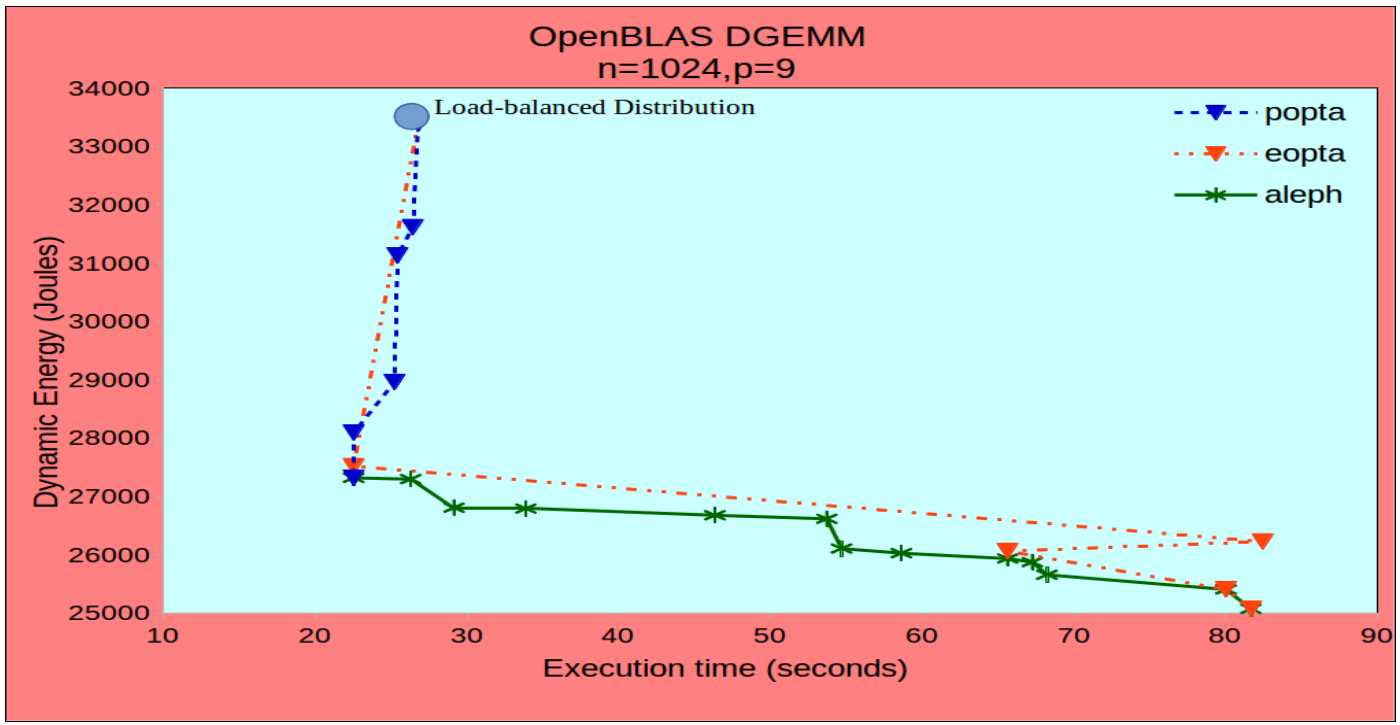
* R. Reddy and A. Lastovetsky, "Bi-Objective Optimization of Data-Parallel Applications on Homogeneous Multicore Clusters for Performance and Energy", IEEE Transactions on Computers, vol. PP, issue 99: IEEE, 08/2017.

Bi-objective vs single-objective optimization



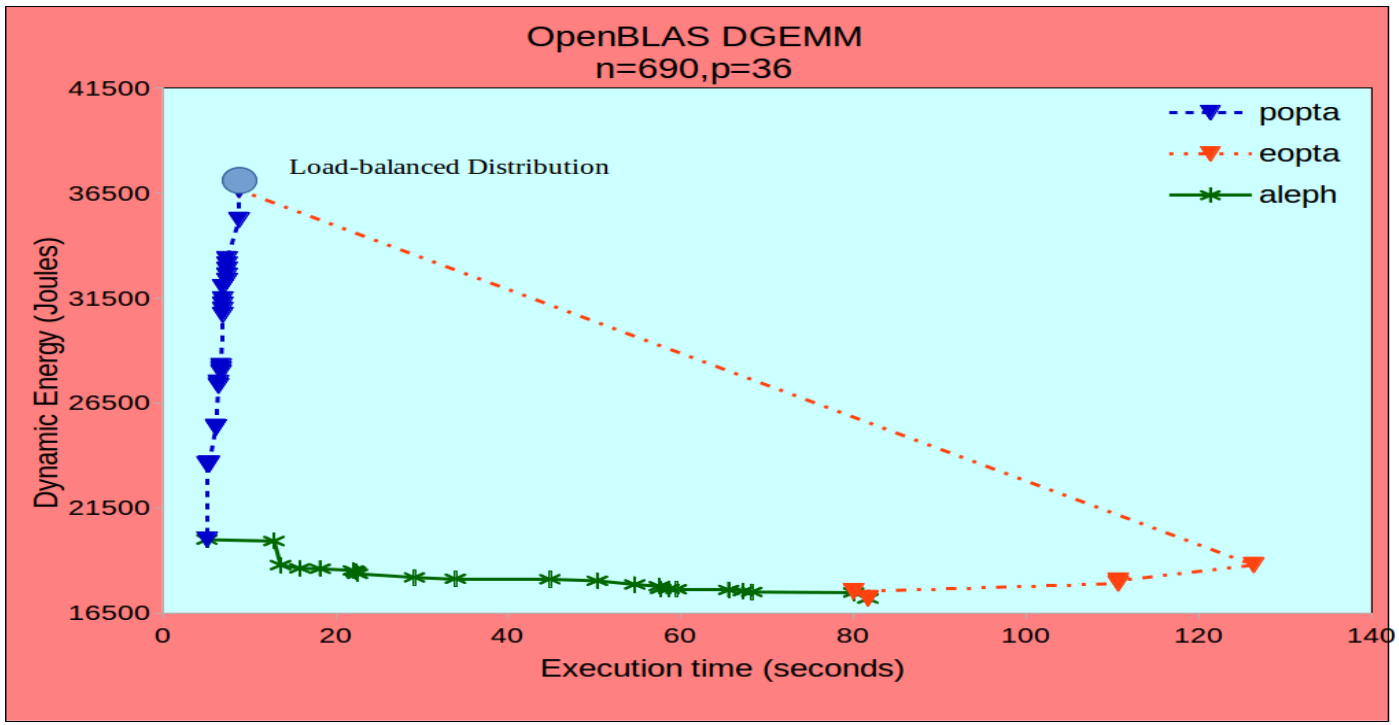
* R. Reddy and A. Lastovetsky, "Bi-Objective Optimization of Data-Parallel Applications on Homogeneous Multicore Clusters for Performance and Energy", IEEE Transactions on Computers, vol. PP, issue 99: IEEE, 08/2017.

Bi-objective vs single-objective optimization



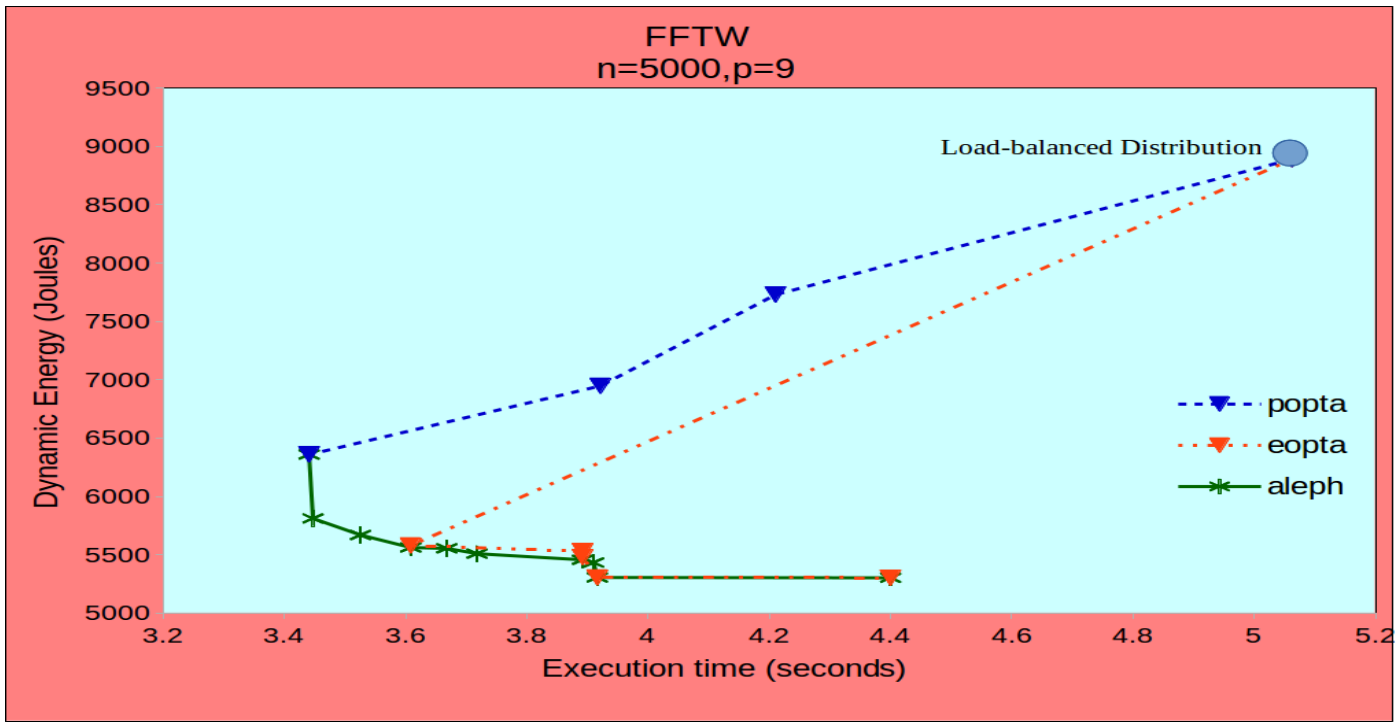
* R. Reddy and A. Lastovetsky, "Bi-Objective Optimization of Data-Parallel Applications on Homogeneous Multicore Clusters for Performance and Energy", IEEE Transactions on Computers, vol. PP, issue 99: IEEE, 08/2017.

Bi-objective vs single-objective optimization



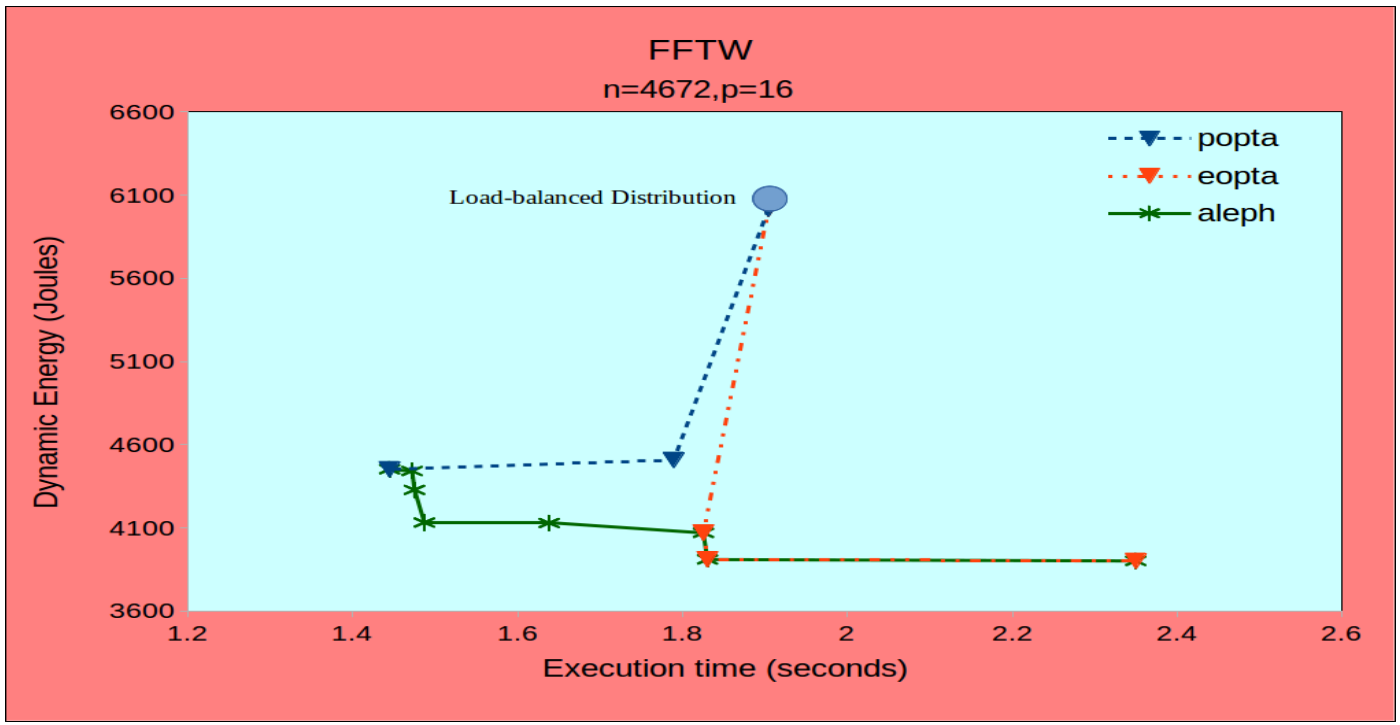
* R. Reddy and A. Lastovetsky, "Bi-Objective Optimization of Data-Parallel Applications on Homogeneous Multicore Clusters for Performance and Energy", IEEE Transactions on Computers, vol. PP, issue 99: IEEE, 08/2017.

Bi-objective vs single-objective optimization



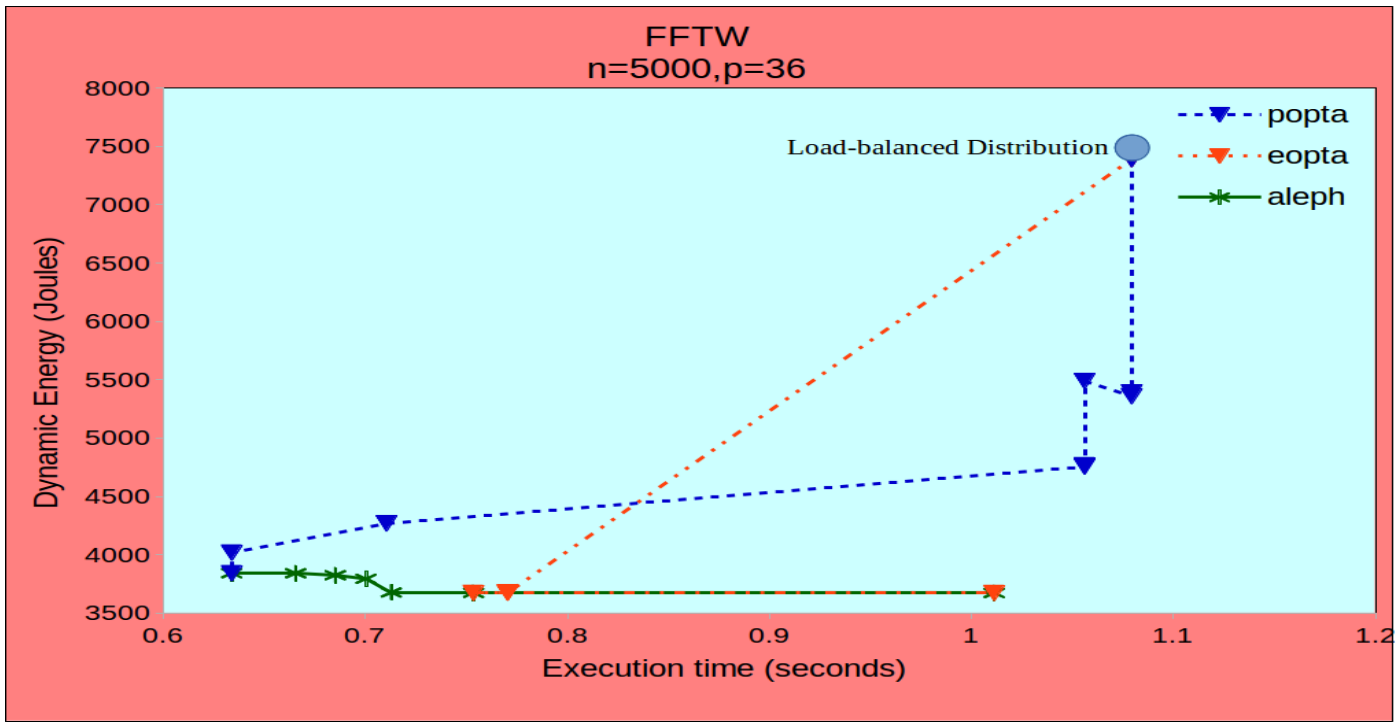
* R. Reddy and A. Lastovetsky, "Bi-Objective Optimization of Data-Parallel Applications on Homogeneous Multicore Clusters for Performance and Energy", IEEE Transactions on Computers, vol. PP, issue 99: IEEE, 08/2017.

Bi-objective vs single-objective optimization



* R. Reddy and A. Lastovetsky, "Bi-Objective Optimization of Data-Parallel Applications on Homogeneous Multicore Clusters for Performance and Energy", IEEE Transactions on Computers, vol. PP, issue 99: IEEE, 08/2017.

Bi-objective vs single-objective optimization



* R. Reddy and A. Lastovetsky, "Bi-Objective Optimization of Data-Parallel Applications on Homogeneous Multicore Clusters for Performance and Energy", IEEE Transactions on Computers, vol. PP, issue 99: IEEE, 08/2017.

Summary

- In multi-core era, load balancing is no longer synonymous to optimization.
- Load distribution becomes an important decision variable even for homogeneous multiprocessors.

Acknowledgments

The research was conducted in collaboration with:

Lukasz Szustak, Czestochowa University of Technology
Roman Wyrzykowski, Czestochowa University of Technology,
Ravi Reddy, University College Dublin

Thank You!

Questions?