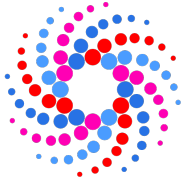
OpenMP®

SC22 Booth Talk Series



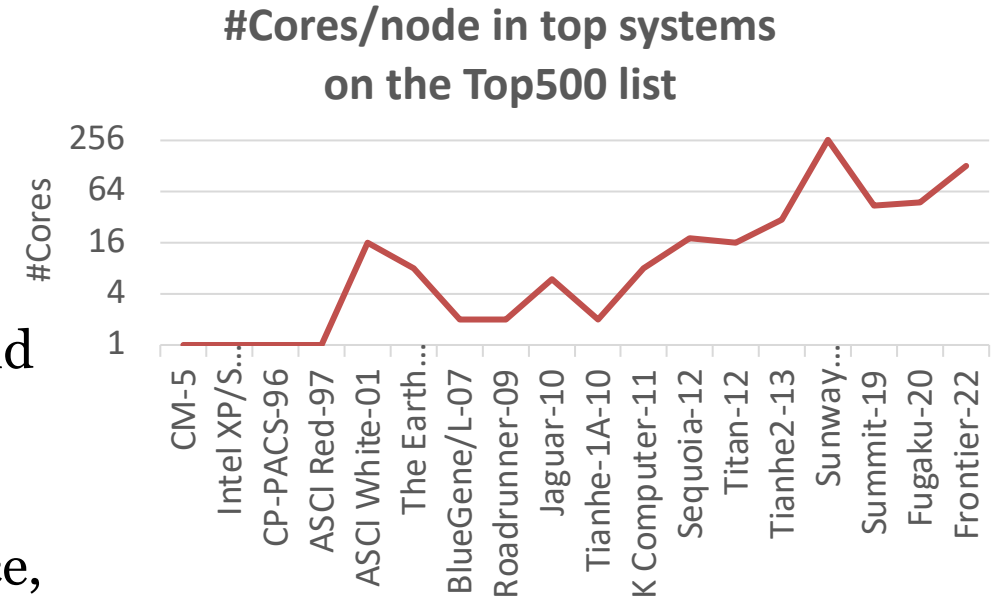
Automated Scheduling Algorithm Selection in OpenMP

http://github.com/unibas-dmi--hpc/LB4OMP

Ali Mohammed[†], Jonas H. Müller Korndörfer[‡], Ahmed Eleliemy[‡], and Florina M. Ciorba[‡]
[†]HPE's HPC/AI EMEA Research Lab (ERL), Switzerland
[‡]Department of Mathematics and Computer Science, University of Basel, Switzerland

Load Imbalance and Impact on Performance

- Increasing #cores/node
- Increased complexity to parallelize and exploit available compute power
- Load imbalance degrades performance, scalability, and wastes energy



Scheduling of Worksharing Loops in OpenMP

static(default)
dynamic runtime
guided auto

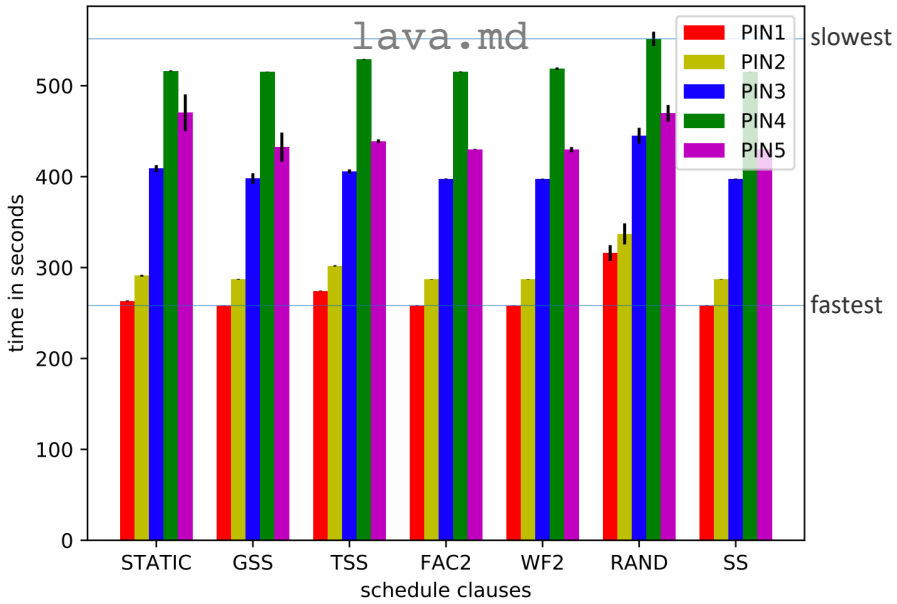
#iteration assigned/round → static/dynamic
threshold for #iteration assigned → guided

Default chunk
static: N/P
dynamic: 1
guided: 1

```
#pragma omp parallel for schedule(kind, chunk)
for(i = 0; i < size; i++)
{
    computations ...
}
```

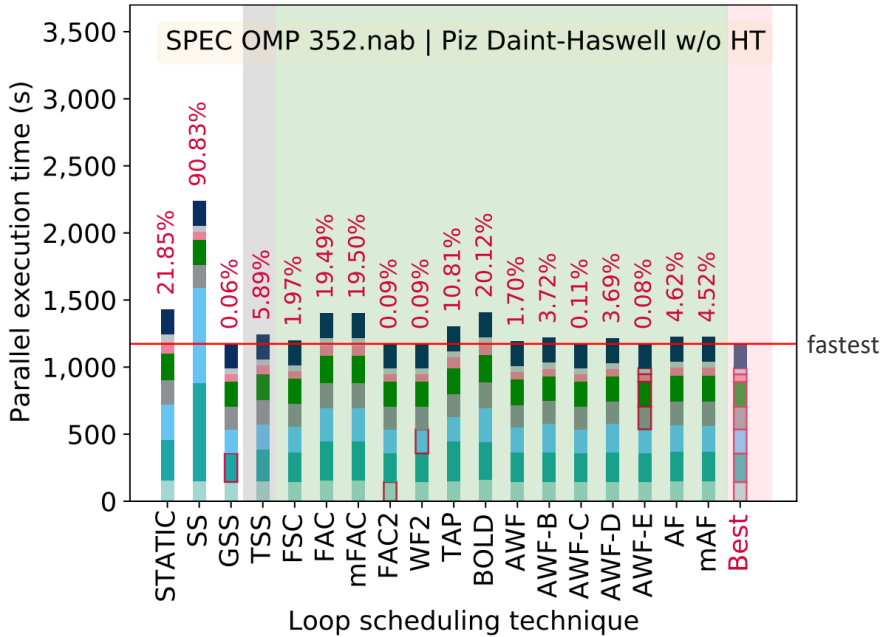
N: #iterations
P: #threads

More Schedules in OpenMP



F. M. Ciorba, C. Iwainsky, and P. Buder, "**OpenMP Loop Scheduling Revisited: Making a Case for More Schedules**", in Proceedings of the 2018 International Workshop on OpenMP (iWomp 2018), Barcelona, Spain, September 21-23, 2018.

https://arxiv.org/abs/1809.03188



J. H. Müller Korndörfer, A. Eleliemy, A. Mohammed, F. M. Ciorba. "**LB4OMP: A Dynamic Load Balancing Library for Multithreaded Applications**". Transactions on Parallel and Distributed Systems (TPDS), Tier **A***, August 2021.

https://ieeexplore.ieee.org/document/9524500

“Decision Paralysis”

- (Too?) Many choices for `OpenMP schedule(kind)`
- (Way?) Too many possible values for `OpenMP chunk`
- Scheduling choice needed per loop, per time-step, per application, and per system tuples

```
#pragma omp parallel for schedule(auto)
for(i = 0; i < size; i++)
{
    computations ...
}
```

The schedule kind **auto** allows* an OpenMP implementation to choose any possible mapping of iterations in a loop construct to threads in the team.

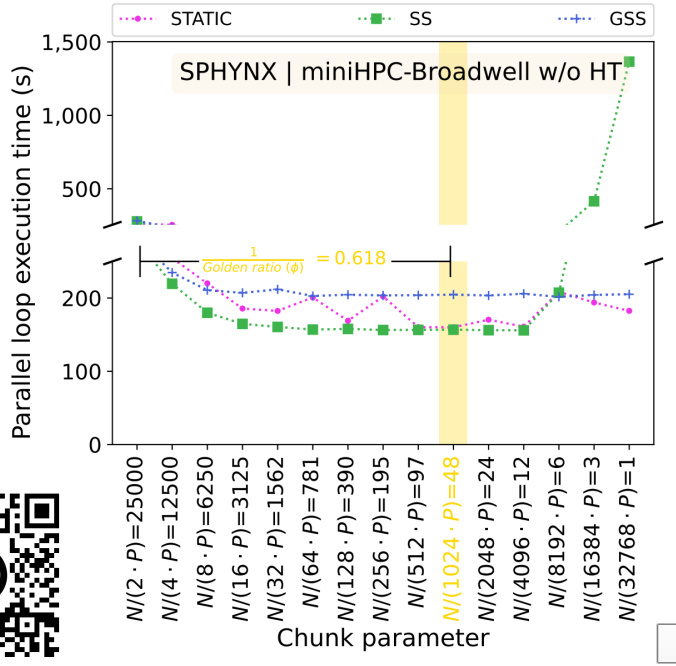
* According to OpenMP Specification 5.2

Proposed Approach: Auto4OMP

http://github.com/unibas-dmi--hpc/LB4OMP

- Leverages **`auto`** as a scheduling option for `schedule(kind)`
- 3 scheduling algorithm selection methods
  - `RandomSel`
  - `ExhaustiveSel`
  - `ExpertSel`
- **Selects** from a portfolio of scheduling algorithms
  - STATIC, SS, GSS, GAC, TSS, Static Steal, mFAC2, AWF-B, AWF-C, AWF-D, AWF-E, mAF

- Extends the LLVM OpenMP runtime library with the `expert chunk parameter`

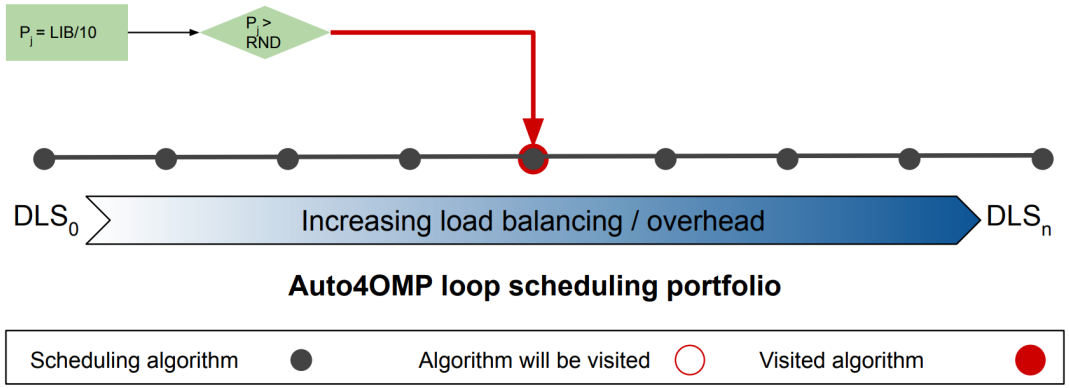


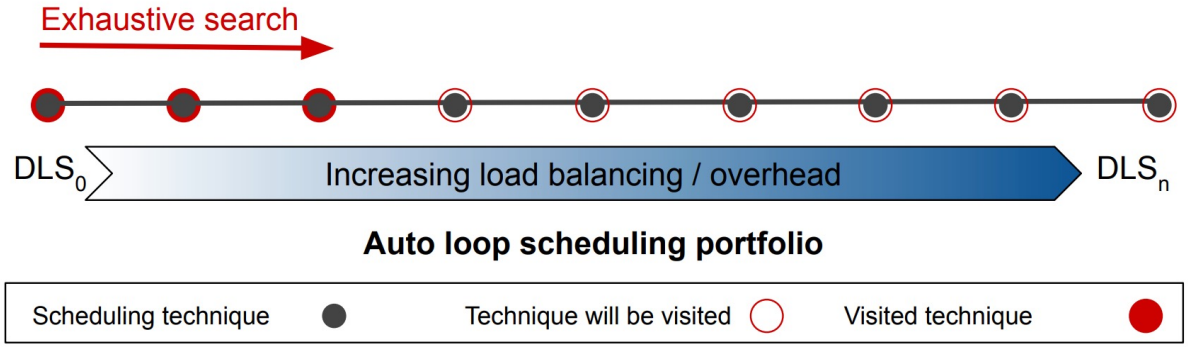
A. Mohammed, J. H. M. Korndörfer, A. Eleliemy and F. M. Ciorba, "**Automated Scheduling Algorithm Selection and Chunk Parameter Calculation in OpenMP**," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 4383-4394, 1 Dec. 2022, DOI: 10.1109/TPDS.2022.3189270, Open Access.
https://ieeexplore.ieee.org/document/9825675/

Auto4OMP: Scheduling Algorithm Selection Methods

RandomSel

- Random selection
- If LIB > 10 changes selected technique





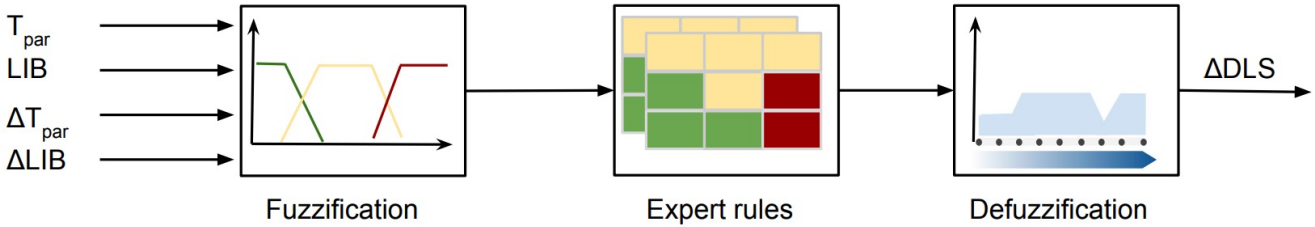
ExhaustiveSel

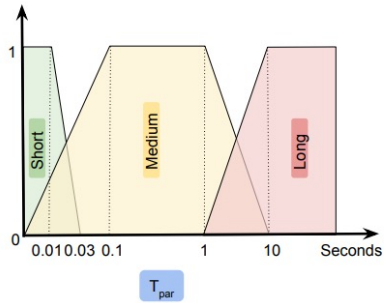
- Exhaustive search
- If LIB > 10, re-evaluate the selected technique

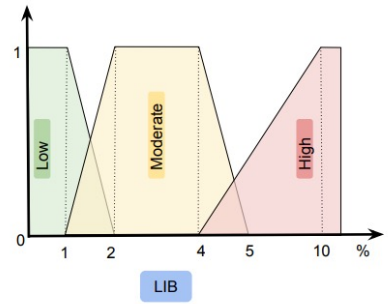
Auto4OMP: Scheduling Algorithm Selection Methods

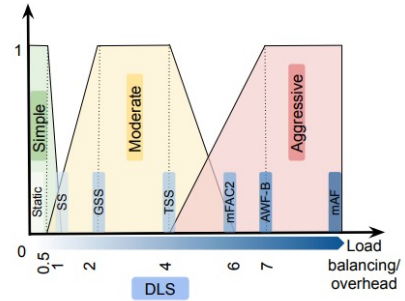
ExpertSel

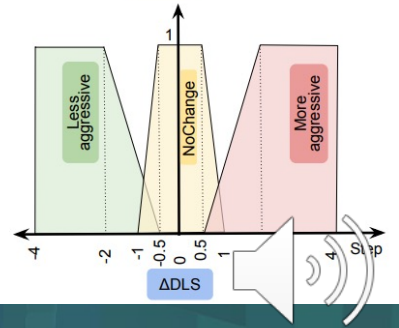
- Employs fuzzy logic
- Constantly evaluates across execution instances
  - Loop execution time
  - Load imbalance
  - Change of loop execution time
  - Change of load imbalance











Expert rules

T_{par} \ LIB	Low	Moderate	High
Short	Simple	Simple	Simple
Medium	Simple	Moderate	Moderate
Long	Simple	Moderate	Aggressive

Auto4OMP: Setup for Experimental Evaluation

Applications

- ALYA (computational mechanics)
- SPEC OMP 2012 352.nab (molecular modeling)
- SPHYNX (astrophysics)
- Mandelbrot (mathematics)
- GROMACS (molecular dynamics)

Algorithm Selection Methods

- LLVM auto (GAC)
- Manual selection
  - ManualBest
  - Oracle (knows all)
- Auto4OMP selection
  - RandomSel
  - ExhaustiveSel
  - ExpertSel

Chunk parameter

- Default chunk
- Expert chunk

Systems

- miniHPC-Broadwell
  - 2 sockets, 10 cores each
- Piz Daint-Haswell
  - 1 socket, 12 cores
- miniHPC-Cascade-Lake
  - 2 sockets, 28 cores each

Experiments

25 loops x 5 applications x 3 systems using x 18 schedule(kinds) x 2 chunk parameters

= **2'700 total experiments**

Evaluation and Results

Hypothesis 1. Auto4OMP achieves high performance and provides the smallest variation of performance across application-system pairs.

Low performance variation across applications and systems

Selection / App-Sys pair	Prior work (no or manual selection)			Auto4OMP				Oracle exec. time
	Static Steal	GAC (LLVM-auto)	ManualBest	SS,expert chunk	RandomSel	ExhaustiveSel	ExpertSel	
ALYA-Piz Daint-Haswell	11.48%	2.44%	2.03%	2.93%	0.29%	0.23%	0.89%	11,492s
GROMACS-miniHPC-Broadwell	22.11%	0.44%	0.11%	1.42%	1.18%	0.56%	0.67%	1,954s
GROMACS-Piz Daint-Haswell	52.12%	0.27%	0.17%	5.26%	0.75%	0.49%	1.10%	22,248s
Mandelbrot-miniHPC-Broadwell	0.36%	0.38%	0.14%	0.02%	0.54%	0.36%	0.87%	1,503s
Mandelbrot-Piz Daint-Haswell	0.20%	0.37%	0.09%	0.13%	0.41%	0.37%	0.45%	8,394s
Mandelbrot-miniHPC-Cascade-Lake	0.46%	0.45%	0.35%	0.31%	1.20%	1.64%	1.42%	1,503s
SPEC OMP 2012 352.nab-miniHPC-Broadwell	4.64%	3.08%	0.20%	2.91%	2.83%	0.96%	2.42%	825s
SPEC OMP 2012 352.nab-Piz Daint-Haswell	3.84%	2.33%	1.59%	2.58%	2.30%	1.99%	2.13%	1,191s
SPHYNX-miniHPC-Broadwell	22.01%	10.36%	5.43%	0.02%	13.48%	1.58%	0.55%	3,863s
SPHYNX-Piz Daint-Haswell	38.81%	12.38%	3.04%	0.11%	15.89%	1.65%	0.50%	4,918s
SPHYNX-miniHPC-Cascade-Lake	26.08%	11.02%	2.45%	0.29%	13.71%	1.19%	1.13%	1,321s

High performance variation across applications and systems

Evaluation and Results

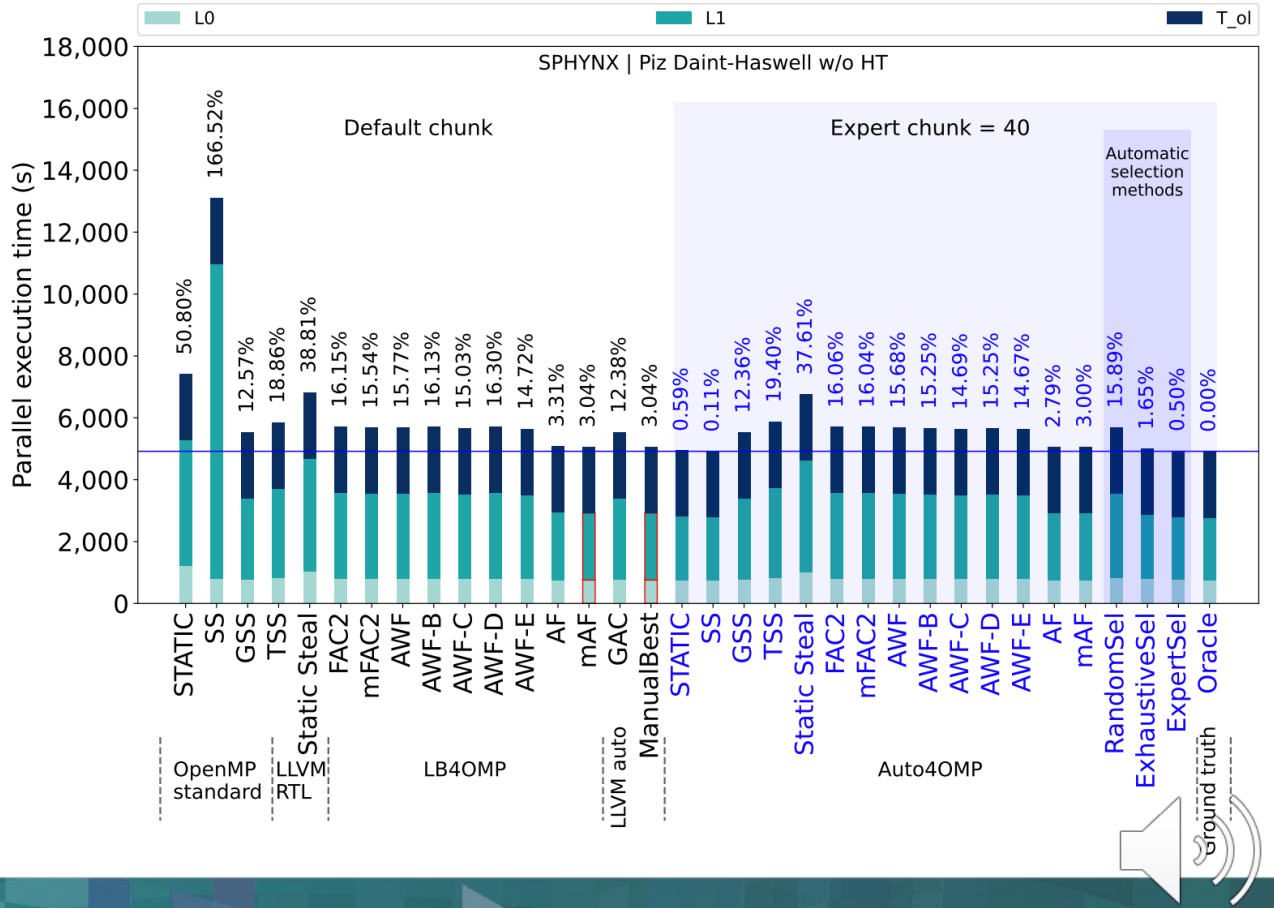
Hypothesis 2. Auto4OMP adapts to the various scheduling needs of applications when they execute on **different systems**.

Selection / App-Sys pair	Prior work (no or manual selection)			Auto4OMP				Oracle exec. time
	Static Steal	GAC (LLVM-auto)	ManualBest	SS,expert chunk	RandomSel	ExhaustiveSel	ExpertSel	
ALYA-Piz Daint-Haswell	11.48%	2.44%	2.03%	2.93%	0.29%	0.23%	0.89%	11,492s
GROMACS-miniHPC-Broadwell	22.11%	0.44%	0.11%	1.42%	1.18%	0.56%	0.67%	1,954s
GROMACS-Piz Daint-Haswell	52.12%	0.27%	0.17%	5.26%	0.75%	0.49%	1.10%	22,248s
Mandelbrot-miniHPC-Broadwell	0.36%	0.38%	0.14%	0.02%	0.54%	0.36%	0.87%	1,503s
Mandelbrot-Piz Daint-Haswell	0.20%	0.37%	0.09%	0.13%	0.41%	0.37%	0.45%	8,394s
Mandelbrot-miniHPC-Cascade-Lake	0.46%	0.45%	0.35%	0.31%	1.20%	1.64%	1.42%	1,503s
SPEC OMP 2012 352.nab-miniHPC-Broadwell	4.64%	3.08%	0.20%	2.91%	2.83%	0.96%	2.42%	825s
SPEC OMP 2012 352.nab-Piz Daint-Haswell	3.84%	2.33%	1.59%	2.58%	2.30%	1.99%	2.13%	1,191s
SPHYNX-miniHPC-Broadwell	22.01%	10.36%	5.43%	0.02%	13.48%	1.58%	0.55%	3,863s
SPHYNX-Piz Daint-Haswell	38.81%	12.38%	3.04%	0.11%	15.89%	1.65%	0.50%	4,918s
SPHYNX-miniHPC-Cascade-Lake	26.08%	11.02%	2.45%	0.29%	13.71%	1.19%	1.13%	1,321s

Low performance variation across systems, relative to Oracle

Evaluation and Results

Hypothesis 3. The use of the **expert chunk parameter** improves application performance at no additional cost.

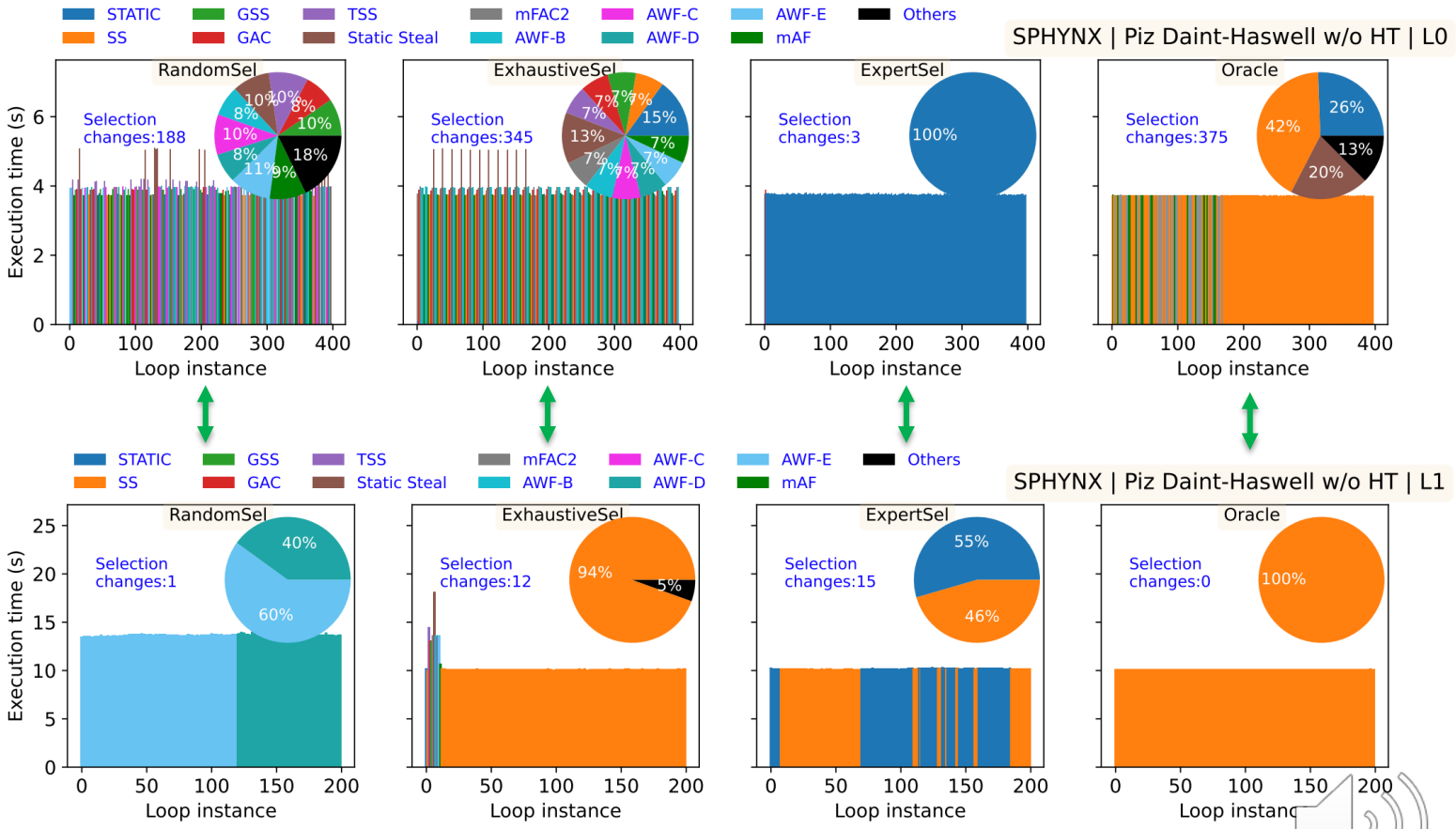


High variation of loop execution time across various scheduling techniques

Evaluation and Results

Hypothesis 4. Auto4OMP adapts to the various scheduling needs of applications' **various loops within a single time-step** (and across time-steps).

Auto4OMP selects different scheduling algorithms for SPHYNX' L0 and L1



Evaluation and Results

Hypothesis 5. Reducing OpenMP thread-level load imbalance improves overall performance of hybrid process+thread parallel applications (MPI+OpenMP).

Improvement over STATIC cumulated over 200 time-steps:

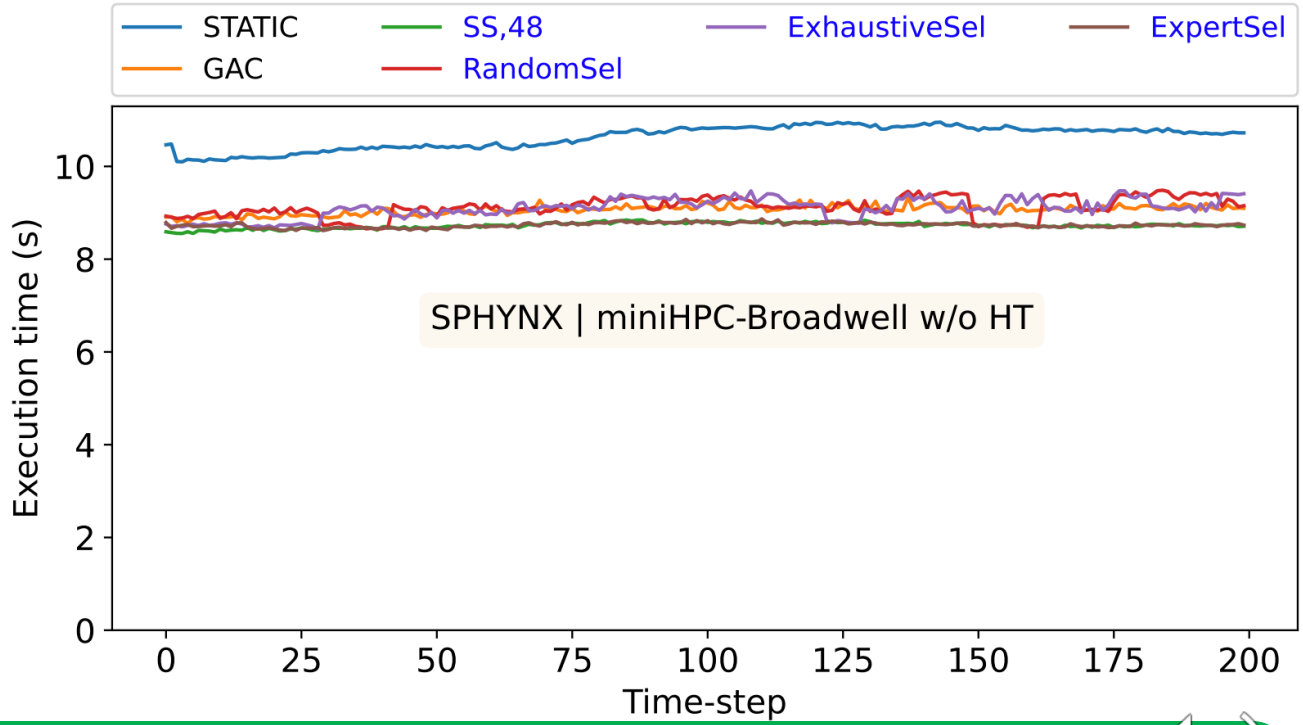
GAC: 17.39%

SS,expert chunk parameter: 22.08%

RandomSel: 21.17%

ExhaustiveSel: 21.65%

ExpertSel: 21.22%



Thread-level load balancing plays a **significant** role in improving the performance of MPI+OpenMP appli ons

Summary

- ManualBest and Oracle require exhaustive offline experimentation
- Auto4OMP is a fully automated load balancing solution for (work sharing) loops in OpenMP for various applications-systems pairs at no additional cost from the user
- Auto4OMP provides:
  - An expert chunk parameter
  - Three scheduling algorithm selection methods
    - RandomSel
    - ExhaustiveSel
    - ExpertSel
- Up to 11% performance improvement over current schedule *auto* implementation in LLVM
- First step towards automated load balancing in OpenMP

Next Steps

Automated Scheduling Algorithm Selection for Process-level Load Balancing

MPI+OpenMP

Algorithm Selection During a Single Loop Execution

Expert-based
vs.
Machine-learning-based

More OpenMP constructs

More Scheduling/Selection Methods

Contributions

OpenMP applications can now benefit from an **automated approach** of **selecting** the scheduling algorithm that **fits** the loop, in a given time-step, on a given platform

Download Auto4OMP, use it, and tell us how it works for your application

http://github.com/unibas-dmi--hpc/LB4OMP

OpenMP®

SC22 Booth Talk Series

openmp.org OpenMP API spec, videos, reference guides, and more

link.openmp.org/sc22 Videos and PDFs of OpenMP SC22 presentations