AUG 4, 2020

Argonne NATIONAL LABORATORY

Machine-Learning-Based Automatic Performance Tuning

Prasanna Balaprakash*, Stefan Wild, Paul Hovland, Hal Finkel, Xingfu Wu, Michael Kruse*

Argonne National Lab

Mary Hall

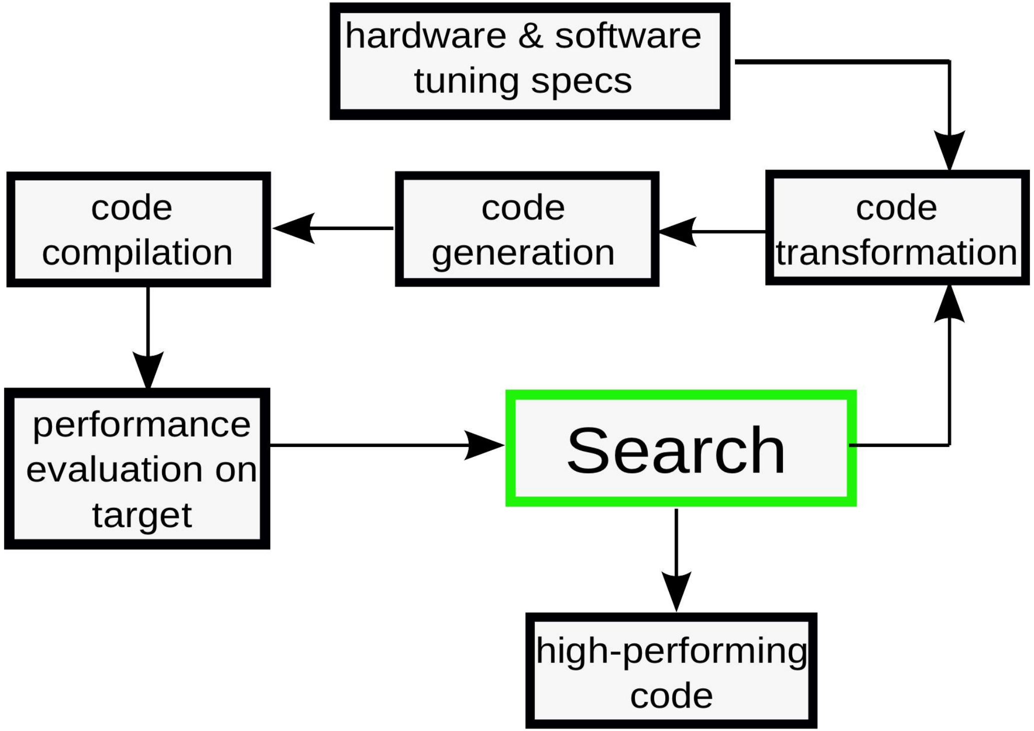
Univ of Utah

EXASCALE COMPUTING PROJECT

U.S. DEPARTMENT OF ENERGY | Office of Science

AUTOMATING EMPIRICAL PERFORMANCE TUNING

For a given application and target platform

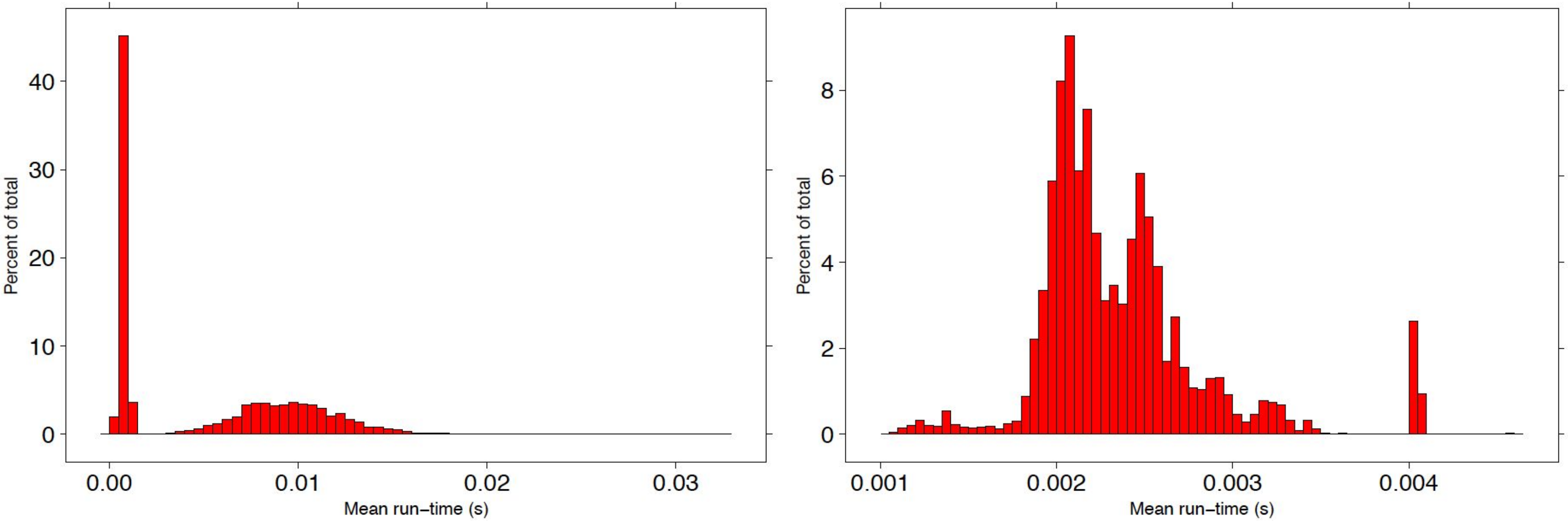


SEARCH IN AUTOTUNING

- Alternatives:
  - Complete enumeration
    - Prohibitively expensive (10^{50} variants!)
    - Unnecessary?
  - Pruning
    - Careful balancing act (between aggressive and conservative)
- Helpful (necessary?) precursors: The expert still plays a role!
  - Identify variable space (parameters to be tuned, ranges, constraints)
  - Quantify measurement limitations and noise
  - Incorporate known models and meaningful objectives

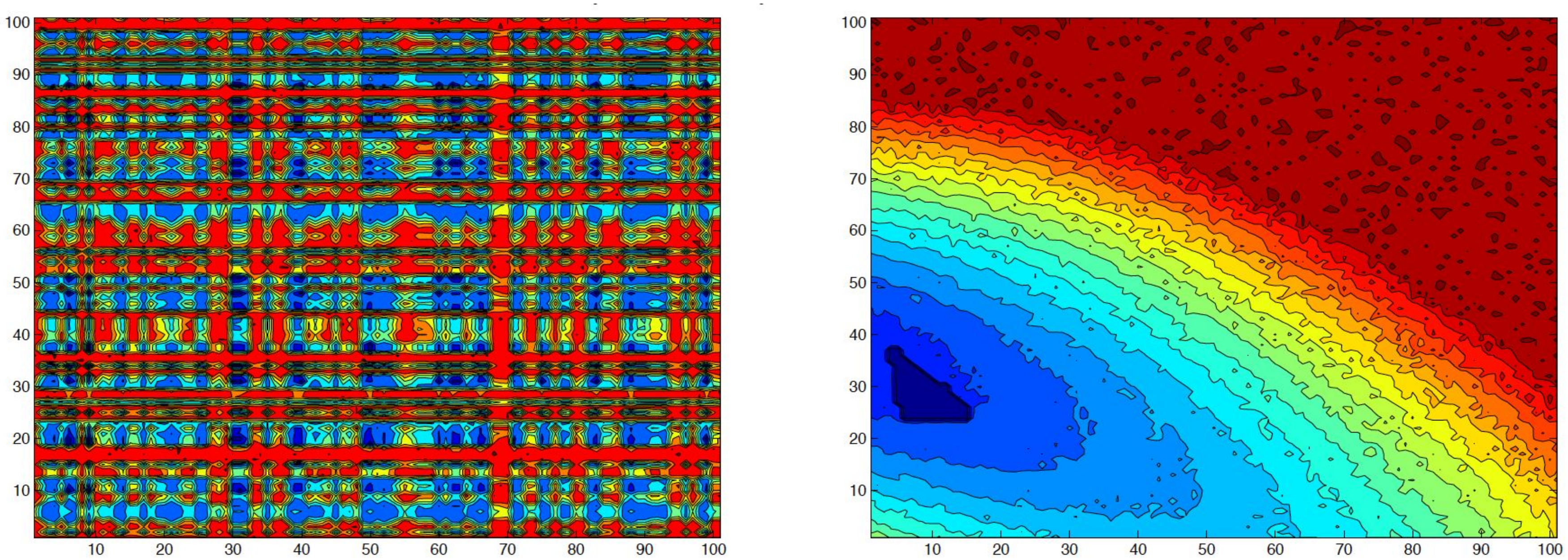
IS A SOPHISTICATED SEARCH ALGORITHM NEEDED?

[Seymour, You, & Dongarra, Cluster Computing ’08]: Random search performs better than tested alternatives as the number of tuning parameters grows



Depends on distribution of high-performing variants

IS A SOPHISTICATED SEARCH ALGORITHM NEEDED?



Depends on structure of the (modeled) search space

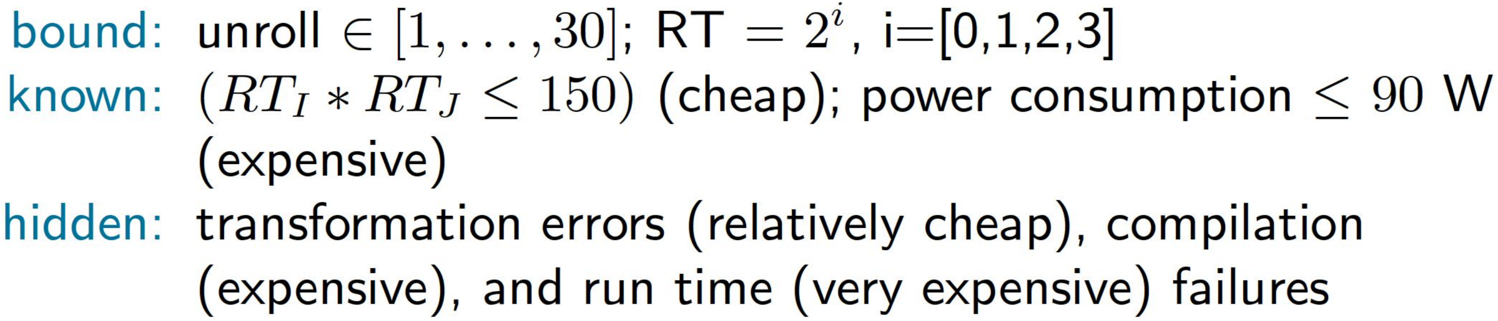
SEARCH AS OPTIMIZATION

Finding the best configuration is a mathematical optimization problem

$$\min_{x} \{ f(x) : x = (x_{\mathcal{I}}, x_{\mathcal{B}}, x_{\mathcal{C}}) \in \mathcal{D} \subset \mathbb{R}^n \}$$

X: multidimensional parameterization (compiler type, compiler flags, unroll/tiling factors, internal tolerances, . . .) for a code variant

f(x): empirical performance metric such as FLOPS, power, or run time (requires a run)



bound: unroll $\in [1, \ldots, 30]$; RT $= 2^i$, i=[0,1,2,3]

known: $(RT_I * RT_J \leq 150)$ (cheap); power consumption ≤ 90 W (expensive)

hidden: transformation errors (relatively cheap), compilation (expensive), and run time (very expensive) failures

OPTIMIZATION CHALLENGES

- Black box, expensive, noisy
- No derivatives
- Discontinuity/unrelaxable parameter values
- Cliffs, multiple local solutions

PREVIOUS AUTOTUNING SEARCH ALGORITHMS

- [Seymour, You, & Dongarra, Cluster Computing ’08] and [Kisuki, Knijnenburg, & O’Boyle, PACT ’00] compared several global and local algorithms
  - Random search outperforms a genetic algorithm, simulated annealing, particle swarm, Nelder-Mead, and orthogonal search
  - Large number of high-performing parameter configurations → easy to find one of them
- [Norris, Hartono, & Gropp, Computational Science ’07] used several global and local algorithms but no comparison
  - Nelder-Mead simplex method, simulated annealing, a genetic algorithm
- Other local search algorithms without comparison to global search:
  - Orthogonal search in ATLAS [Whaley & Dongarra, SC ’98]
  - Pattern search in loop optimization [Qasem, Kennedy & Mellor-Crummey SC ’06]
  - Modified Nelder-Mead simplex algorithm in Active Harmony [Tiwari, Chen, Chame, Hall, & Hollingsworth, IPDPS ’09]

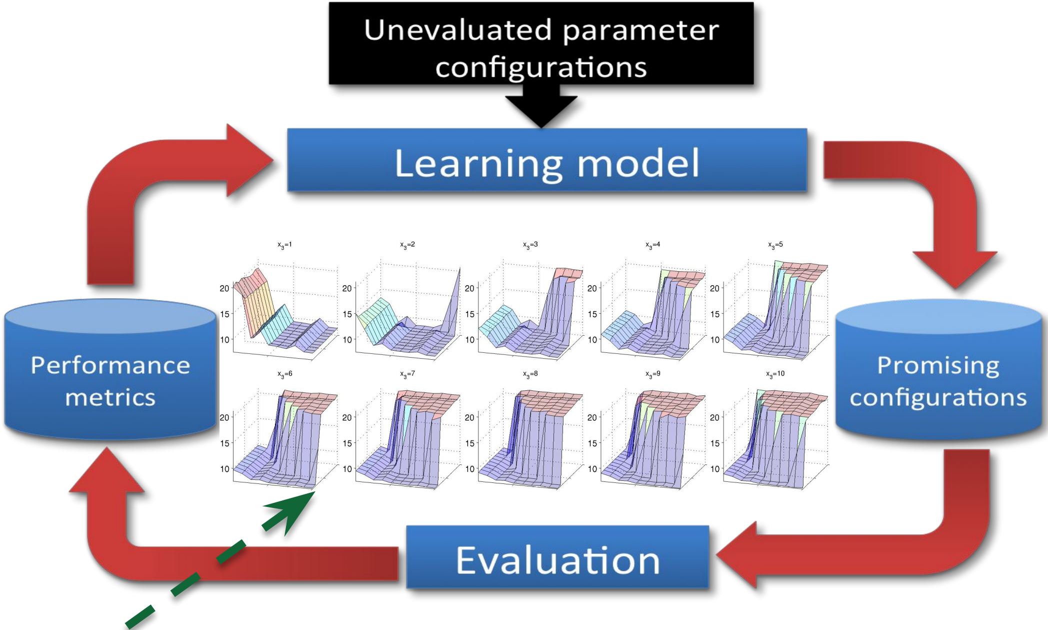
PREVIOUS AUTOTUNING SEARCH ALGORITHMS

- [Seymour, You, & Dongarra, Cluster Computing ’08] and [I [illegible] nburg, & O’Boyle, PACT ’00] compared several global and loc [illegible]
  - Random search outperforms a genetic algori [illegible] annealing, particle swarm, Nelder-Mead, and orthogonal se [illegible]
  - Large number of high-performing pa [illegible] gurations → easy to find one of them
- [Norris, Hartono, & Gropp, Co [illegible] ience ’07] used several global and local algorithms but no compari [illegible]
  - Nelder-Mead sim [illegible] mulated annealing, a genetic algorithm
- Other local sea [illegible] without comparison to global search:
  - Orthogonal [illegible] TLAS [Whaley & Dongarra, SC ’98]
  - Pattern searc [illegible] loop optimization [Qasem, Kennedy & Mellor-Crummey SC ’06]
  - Modified Nelder-Mead simplex algorithm in Active Harmony [Tiwari, Chen, Chame, Hall, & Hollingsworth, IPDPS ’09]

Out-of-the-box not custom

MACHINE-LEARNING BASED SEARCH

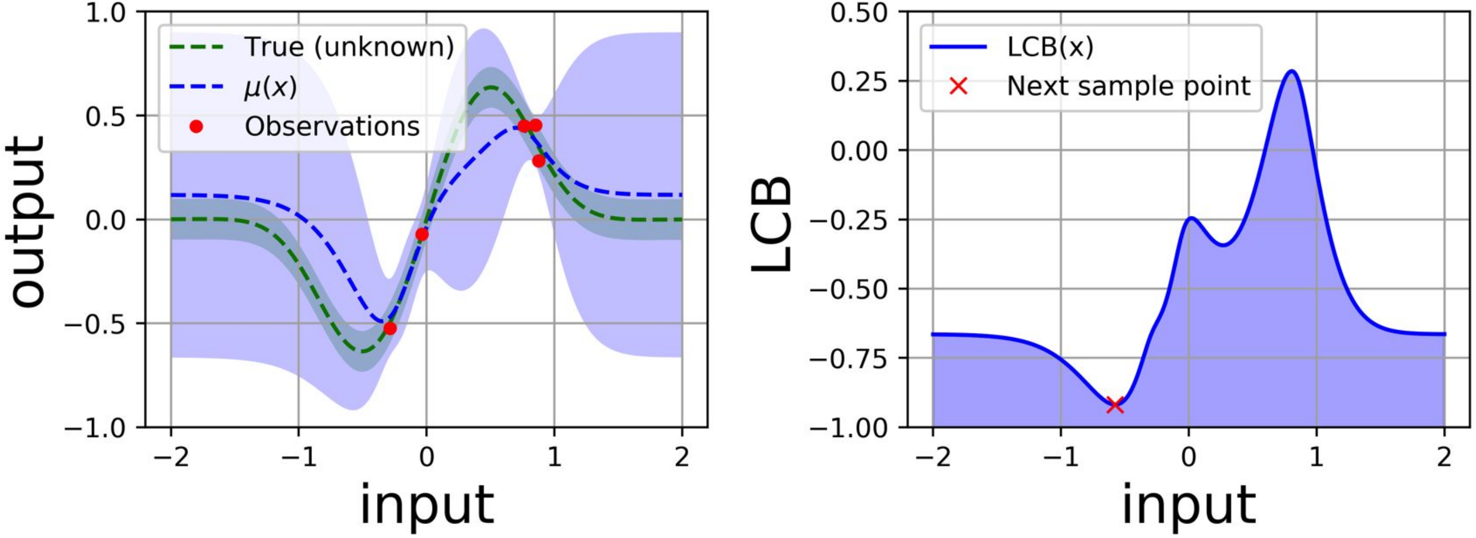
- Framework:
  - Initialization phase
    - Random or Latin hypercube sampling
  - Iterative phase
    - Fit model
    - Sample using the model



Example Surrogate Model Fitted to Sampled Performance (iterative refinement improves the learning model)

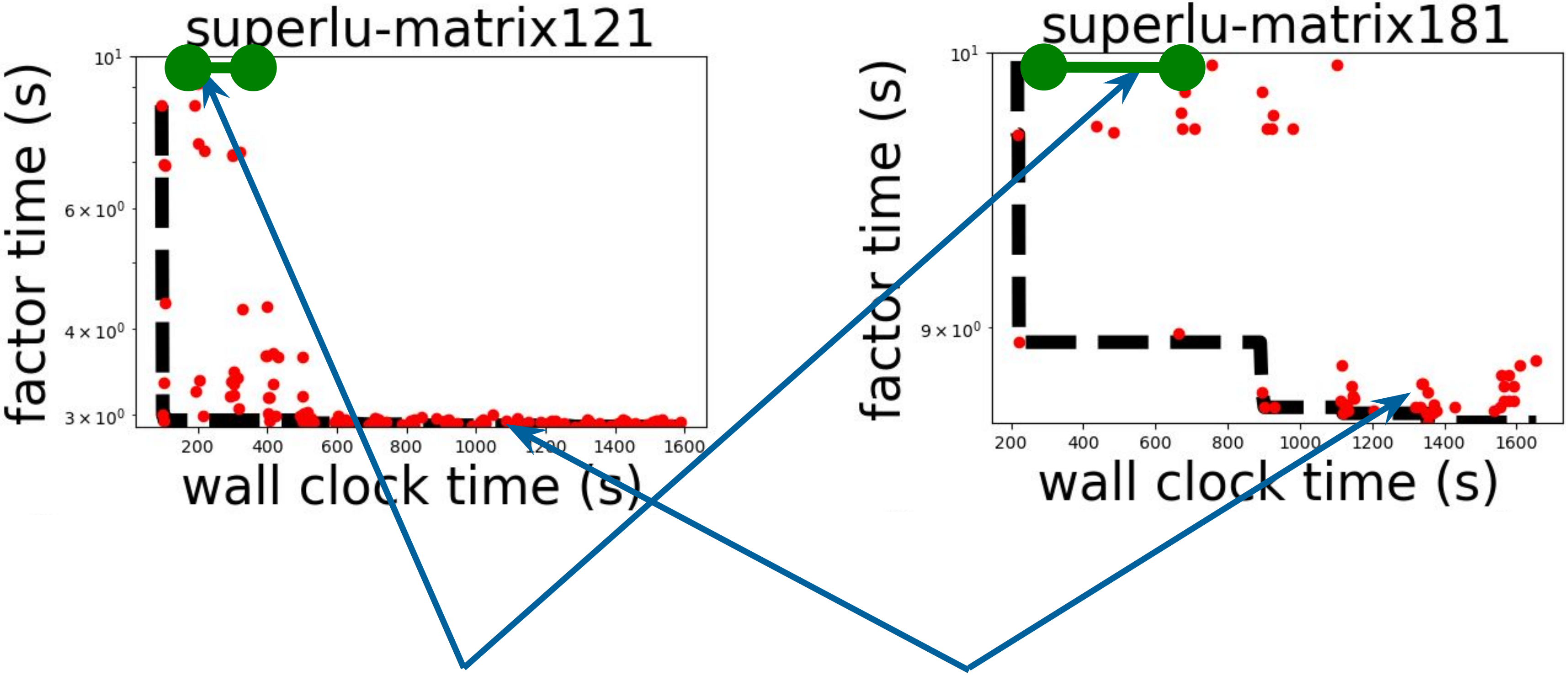
BAYESIAN OPTIMIZATION

$$LCB(x, \beta) = \mu(x) - \beta \times \sigma(x)$$



RESULTS

Edison@Nersc
Each node: 12-core Intel "Ivy Bridge" processor at 2.4 GHz



exploratio

exploitatio

ACKNOWLEDGEMENTS

Exascale computing project

DOE Early Career Research Program, ASCR

REFERENCES

- P. Balaprakash, S. M. Wild, and P. D. Hovland. Can search algorithms save large-scale automatic performance tuning? In Proceedings of the International Conference on Computational Science, ICCS 2011, volume 4, pages 2136–2145, 2011.
- P. Balaprakash, S. M. Wild, and B. Norris. SPAPT: Search Problems in Automatic Performance Tuning. In Proceedings of the International Conference on Computational Science, ICCS 2012, volume 9, pages 1959–1968, 2012.
- P. Balaprakash, S. M. Wild, and P. D. Hovland. An experimental study of global and local search algorithms in empirical performance tuning. In High Performance Computing for Computational Science - VECPAR 2012, 10th International Conference, Revised Selected Papers, Lecture Notes in Computer Science, pages 261–269. Springer, 2013.
- T. Nelson, A. Rivera, P. Balaprakash, M. Hall, P. D. Hovland, E. Jessup, and B. Norris. Generating efficient tensor contractions for GPUs. In 2015 44th International Conference on Parallel Processing (ICPP), pages 969–978, 2015.
- P. Balaprakash, J. Dongarra, T. Gamblin, M. Hall, J. K. Hollingsworth, B. Norris, and R. Vuduc. Autotuning in high-performance computing applications. Proceedings of the IEEE, pages 1–16, 2018.

THANK YOU!

https://github.com/ytopt-team/ytopt