LlamaTune: Sample-Efficient DBMS Configuration Tuning

Konstantinos Kanellis, Cong Ding, Brian Kroth,
Andreas Müller, Carlo Curino, Shivaram Venkataraman

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

Microsoft
Gray Systems Lab

DBMS Configuration Tuning

Tuning DBMS knob values is *essential* for achieving high-performance

default values are sub-optimal; often chosen for compatibility rather than performance

```
...
bgwriter_lru_maxpages: 100
commit_delay: 0
deadlock_timeout: 1000ms
default_statistics_target: 100
effective_cache_size: 4GB
random_page_cost: 1.0
wal_sync_method: fdatasync
work_mem: 4MB
...
```

Default Configuration

PostgreSQL

Tuning Process →

```
...
bgwriter_lru_maxpages: 20
commit_delay: 150
deadlock_timeout: 4500ms
default_statistics_target: 30
effective_cache_size: 16GB
random_page_cost: 1.25
wal_sync_method: open_fdatasync
work_mem: 16MB
...
```

Tuned Configuration

PostgreSQL

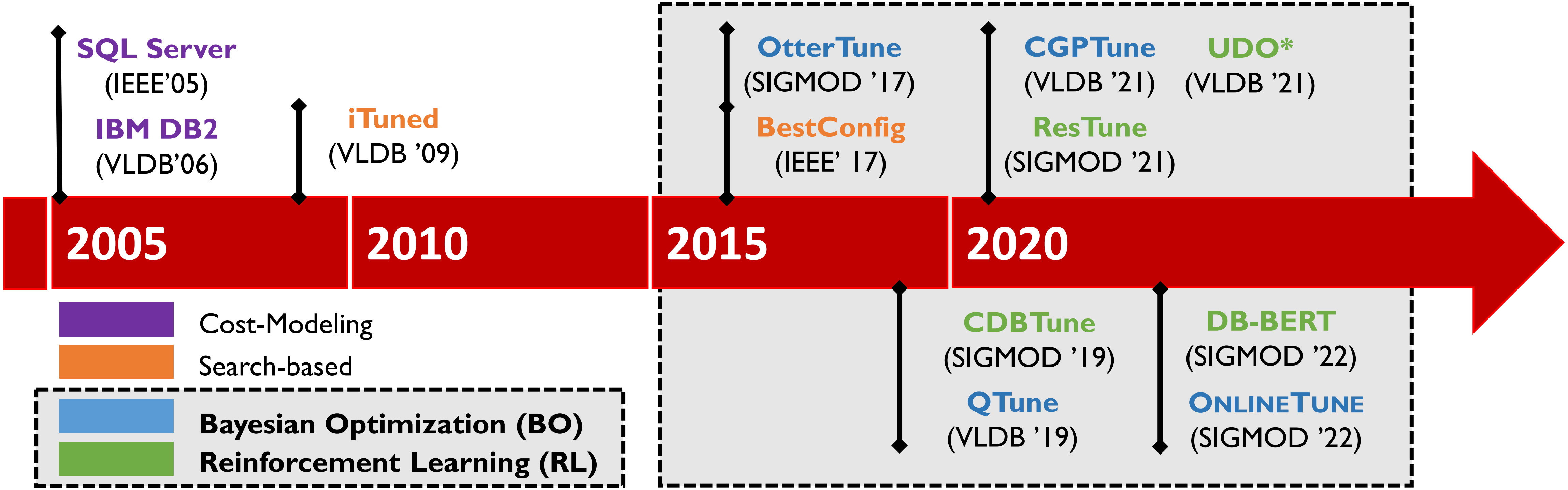
Properly tuned database systems can achieve 3-6x *higher* throughput [1]

[1] Dana Van Aken et. al. *Automatic Database Management System Tuning Through Large-scale Machine Learning*. (SIGMOD ’17)

Automated DBMS Configuration Tuning

DBMSs increasing *complexity* made this task harder for DBAs

- Hundreds of configuration knobs
- Knob heterogeneity (discrete, continuous, categorical)
- Unknown interactions among different knobs (and their values…)



Motivation

Sample-efficiency is a *crucial* requirement to use tuners on *diverse* workloads

Even state-of-the-art optimizers need **~100** samples (>10 hours) to converge to optimal config, when tuning new workloads *without* any prior knowledge

Recent DBMS configuration tuning experimental study / comparison [2]

#1: SMAC the best overall (BO-based \w Random Forest model)

#2: DDPG performed good when tuning ~20 knobs (RL-based, CDBTune)

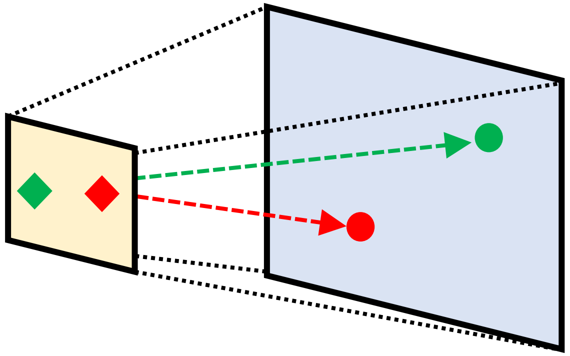
High-dimensional configuration search space is a *major* contributing factor

Can we leverage DBMS-specific *insights* to improve tuning performance?

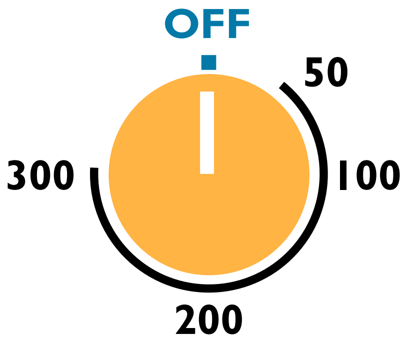
[2] Zhang et. al. Facilitating Database Tuning with Hyper-Parameter Optimization: A Comprehensive Experimental Evaluation (VLDB '22)

LlamaTune Overview

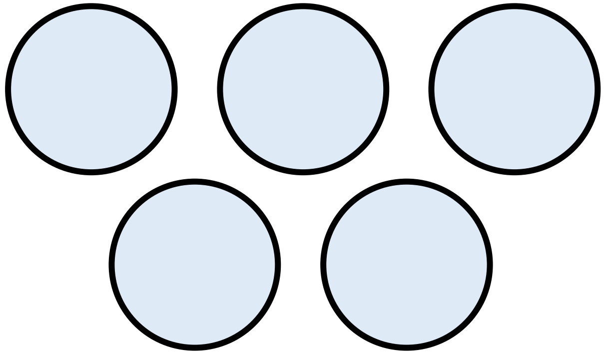
Tuner design that leverages *domain knowledge* to improve the sample efficiency of the underlying configuration optimizers



Low-Dimensional Search Space Tuning



Special Knob Values Handling

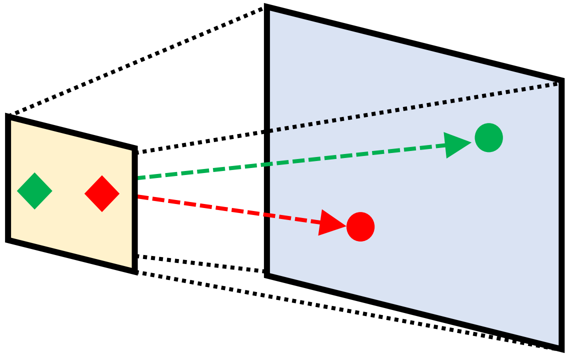


Knob Values Bucketization

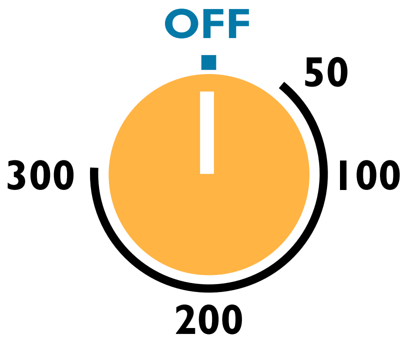
Reduces configuration *evaluations* by up to **11x** ; up to **21%** higher *throughput*

LlamaTune Overview

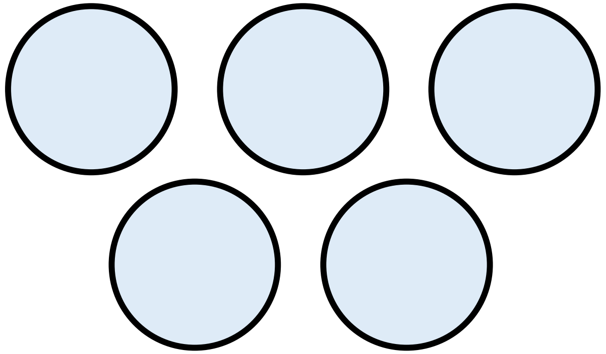
Tuner design that leverages *domain knowledge* to improve the sample efficiency of the underlying configuration optimizers



Low-Dimensional Search Space Tuning



Special Knob Values Handling



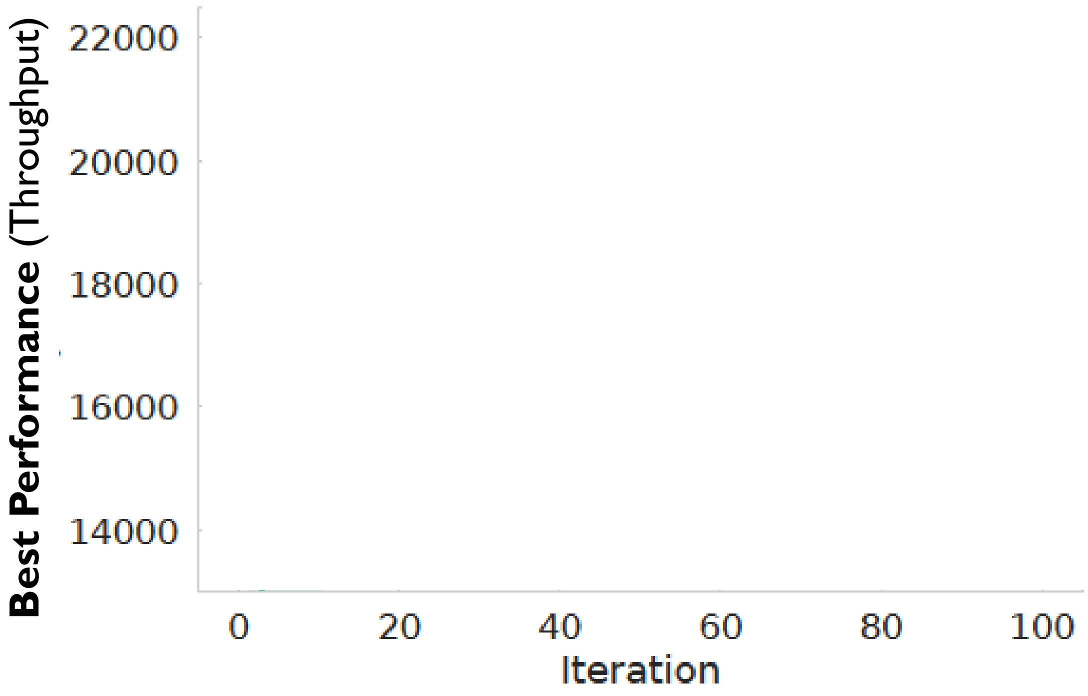
Knob Values Bucketization

Reduces configuration *evaluations* by up to **11x** ; up to **21%** higher *throughput*

Low-Dimensional Tuning – Important Knobs

Tuning few important knobs can yield optimal DBMS performance

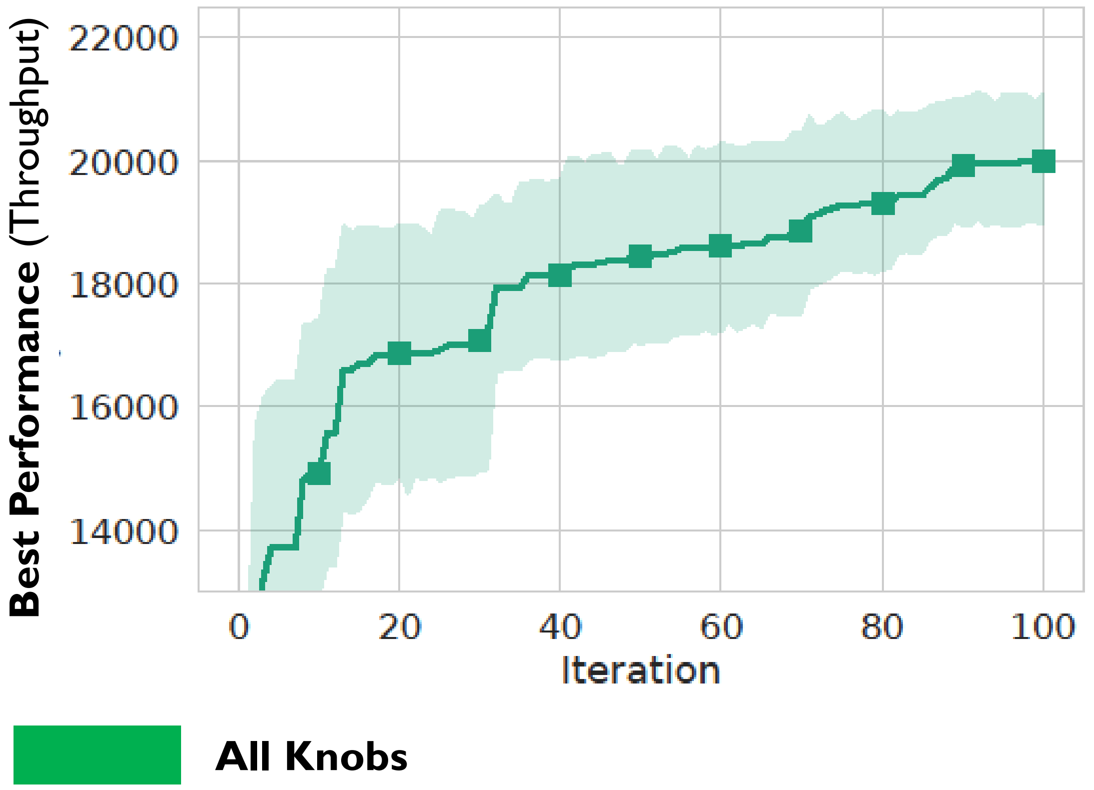
PostgreSQL, 90 knobs, YCSB-A (50%/50% read/write), SF=20, SMAC, average of 5 runs



Low-Dimensional Tuning – Important Knobs

Tuning few important knobs can yield optimal DBMS performance

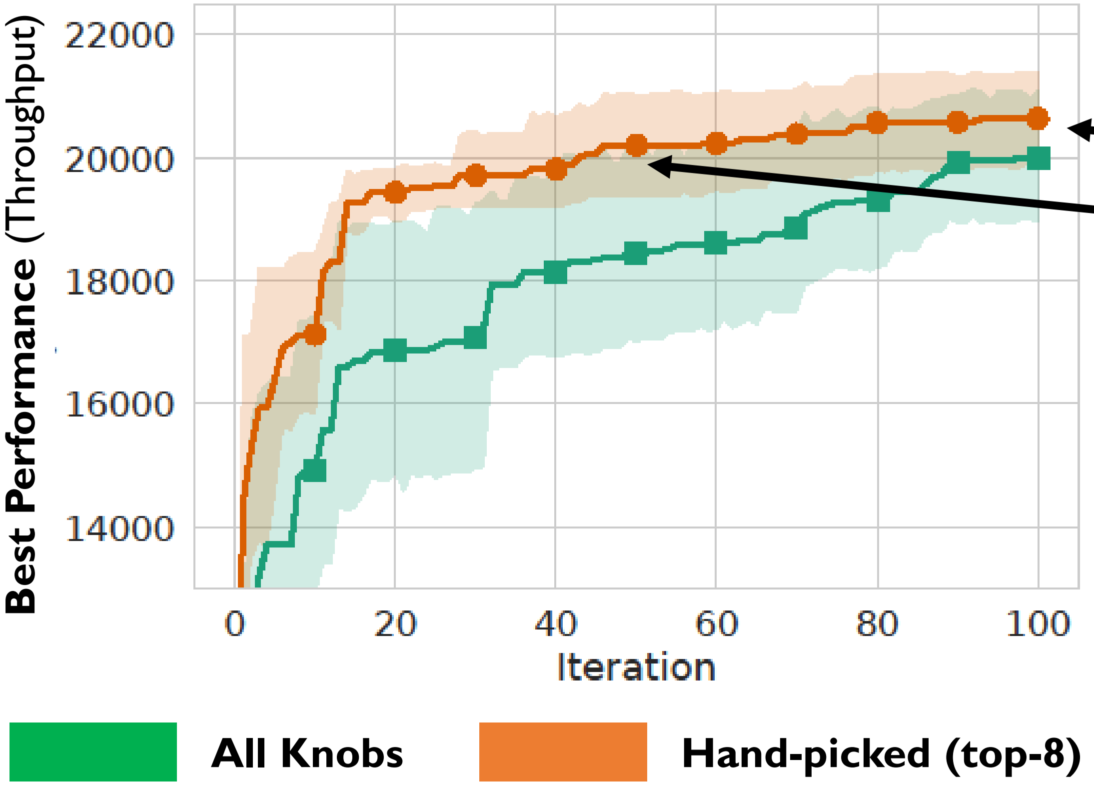
PostgreSQL, 90 knobs, YCSB-A (50%/50% read/write), SF=20, SMAC, average of 5 runs



Low-Dimensional Tuning – Important Knobs

Tuning few important knobs can yield optimal DBMS performance

PostgreSQL, 90 knobs, YCSB-A (50%/50% read/write), SF=20, SMAC, average of 5 runs



Higher Final Throughput

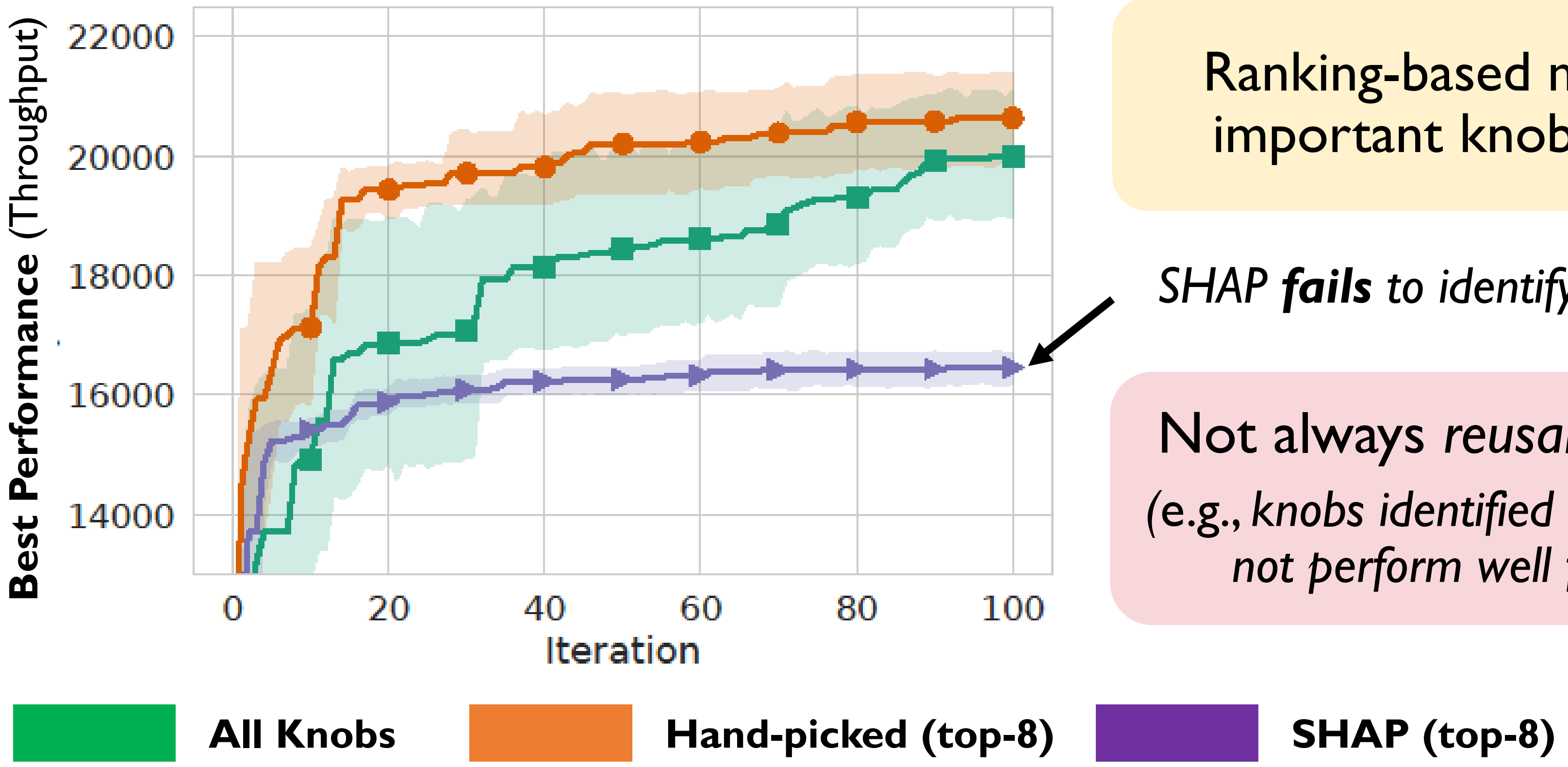
Faster Convergence to the optimal config. (evaluate fewer configurations to reach baseline optimal performance)

Tuning *smaller* search space (8 vs 90 knobs) can realize gains!

Low-Dimensional Tuning – Important Knobs

Tuning few important knobs can yield optimal DBMS performance

PostgreSQL, 90 knobs, YCSB-A (50%/50% read/write), SF=20, SMAC, average of 5 runs



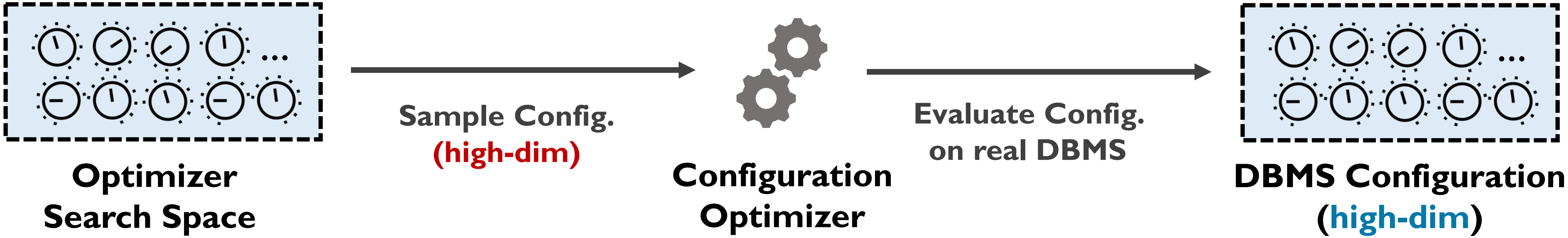
Ranking-based methods that identify important knobs are *slow*, *inaccurate*

*SHAP **fails** to identify <u>all</u> important knobs*

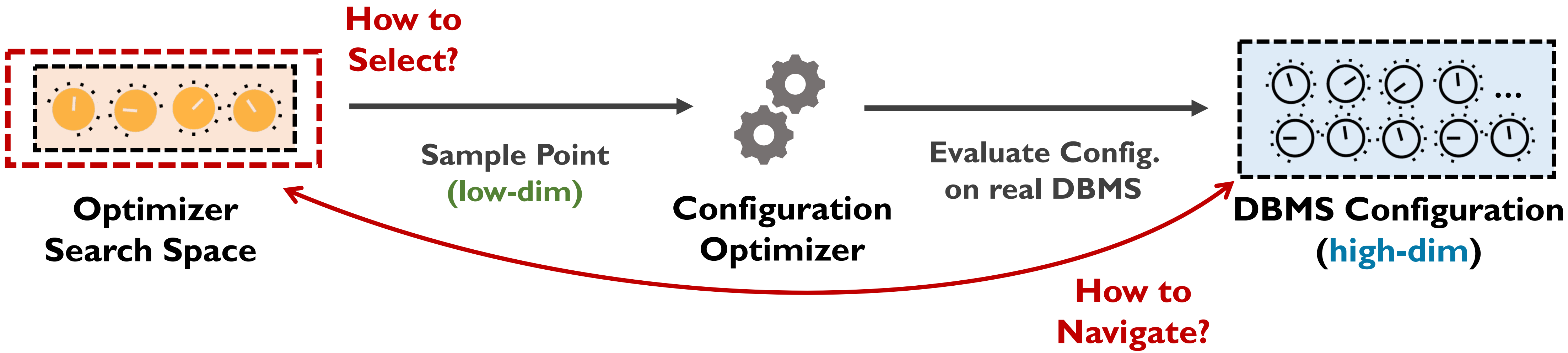
Not always *reusable* across workloads!

(e.g., knobs identified as important for YCSB-A do not perform well for TPC-C, when tuned)

Low-Dimensional Tuning



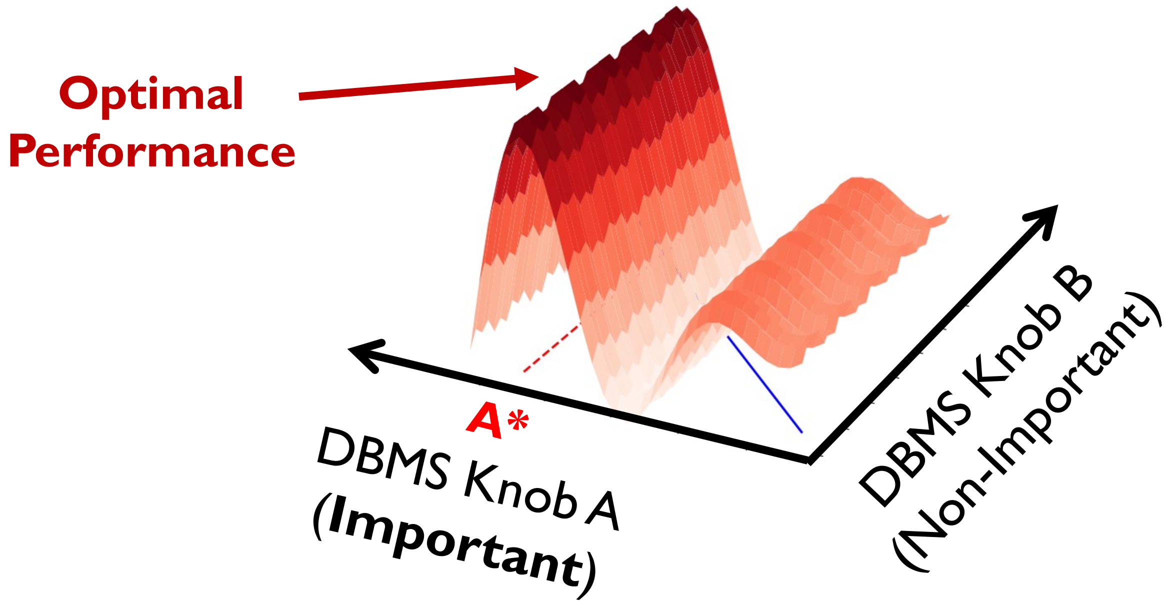
Use fewer knobs (dimensions) to *model* the DBMS performance behavior

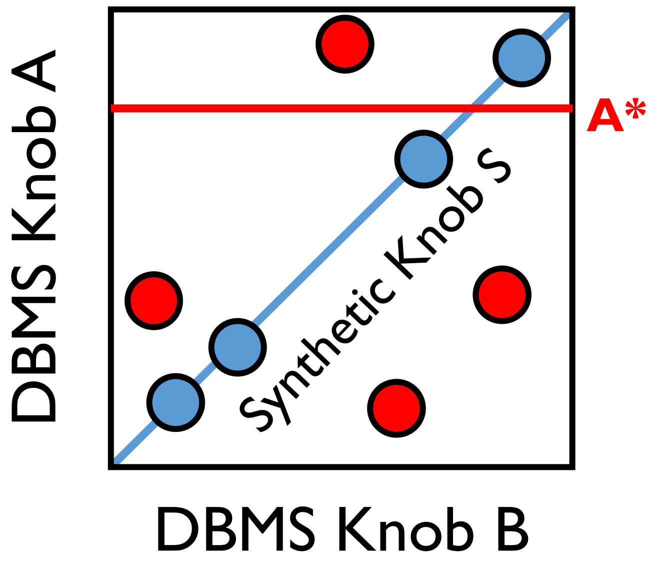


Synthetic Low-Dimensional Search Space

Combine multiple <u>physical</u> DBMS knobs to create few *synthetic* knobs

- No actual meaning themselves – their values determine the **real** DBMS knob values
- Optimizer now tunes these synthetic knobs (i.e., low-dimensional search space)





How to construct this *mapping* from low-dim space to high-dim one?

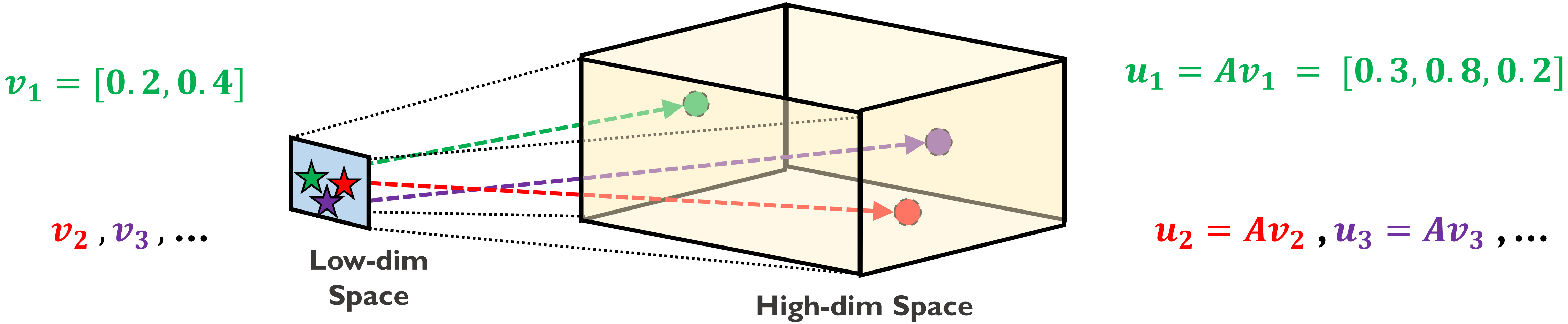
Low-Dimensional Projections

Theoretically-sound proposals from the ML / BO community

- Assume low-dim space has d dimensions; high-dim space has D
- Define a *projection* matrix A, to map points from low-dim to high-dim space

Input: estimate of the *number* of important dimensions (knobs) [d]

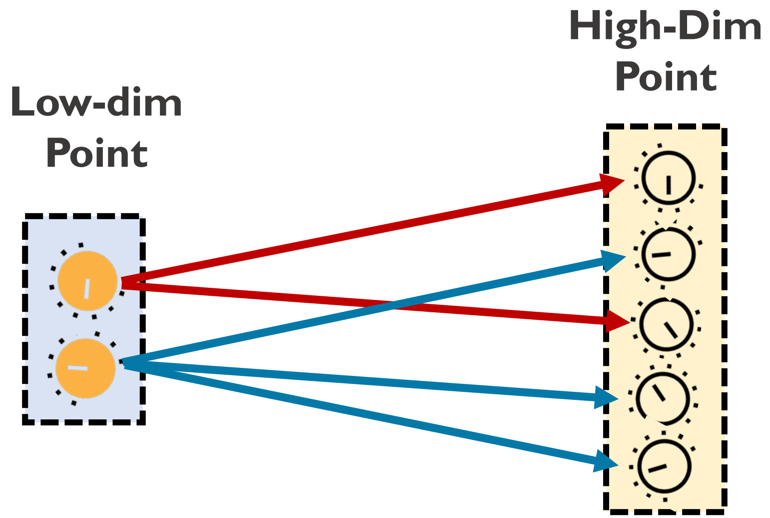
Output: definition of low-dim search space & projection matrix [$d \rightarrow D$]



Low-Dimensional Projections

Hashing-Enhanced Subspace BO (HesBO) [3]

- Random one-to-many *linear* projection
- Based on Count-Sketch projection
- Adequately preserves the characteristics of up to d (important) dimensions (e.g., pairwise distances)

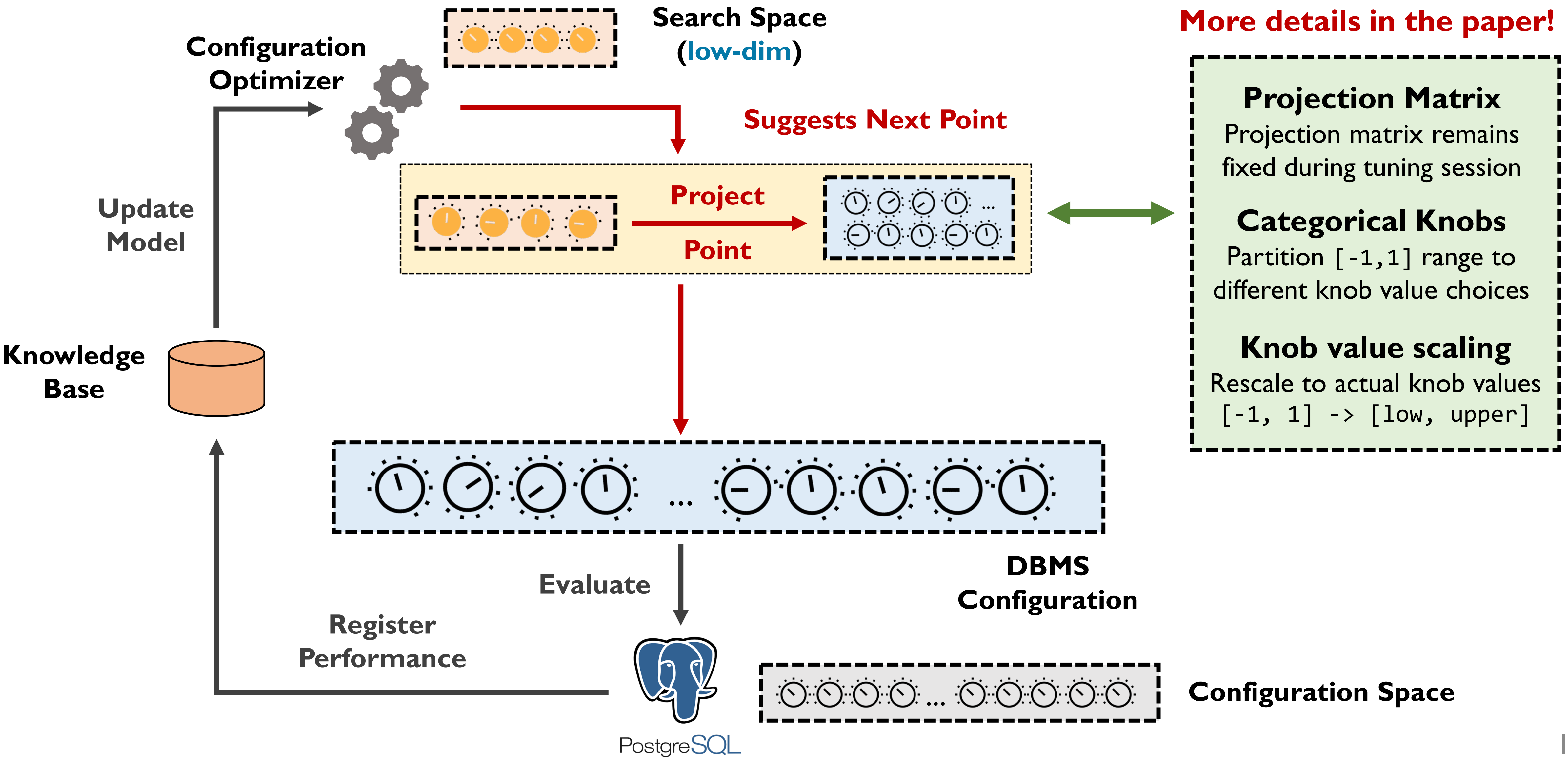


Low-dim space dimensions [d] **>** True # of important dimensions [d_e]

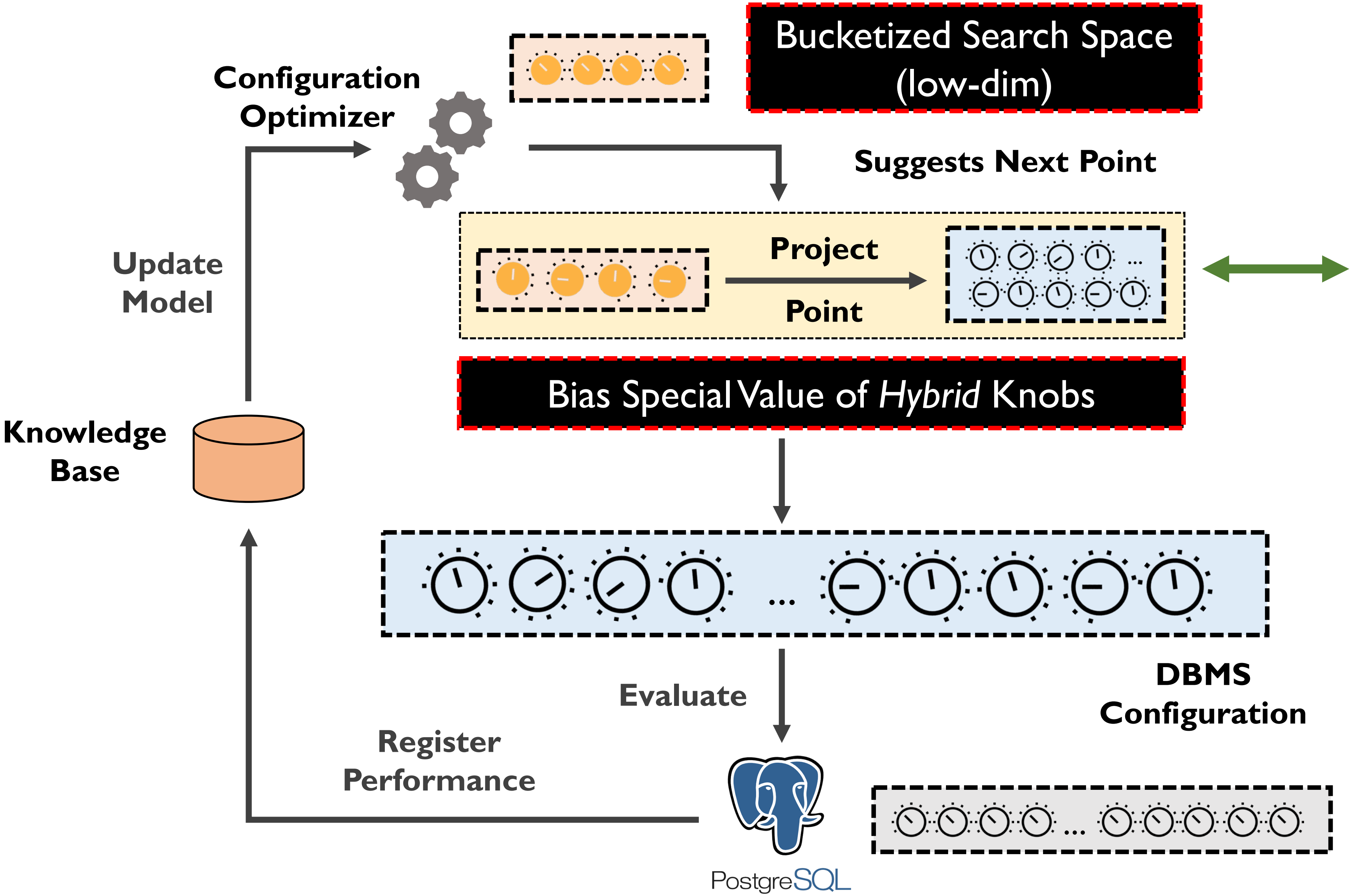
Optimal point(s) in low-dimensional space, with **high-probability**!

[3] Munteanu et al. A Framework for Bayesian Optimization in Embedded Subspaces (PMLR'19)

LlamaTune Architecture (low-dim tuning)



LlamaTune Architecture (complete)



More details in the paper!

Projection Matrix
Projection matrix remains fixed during tuning session

Categorical Knobs
Partition `[-1,1]` range to different knob value choices

Knob value scaling
Rescale to actual knob values
`[-1, 1] -> [low, upper]`

Evaluation

CloudLab

End-to-end evaluation with *six* diverse workloads

- TPC-C, SEATS, Twitter, YCSB-A, YCSB-B, ResourceStresser

Multiple performance *targets*

- Max throughput, 95-th% tail latency

Different underlying configuration optimizer

- SMAC, Gaussian-Based Bayesian Optimizer (GP-BO) ; DDPG (RL-Based)

Generalization to *newer* PostgreSQL version

Ablation Studies

- Measure *how much* each component contributes

Sensitivity analysis for each individual component

Overhead of the configuration optimizer

Evaluation

CloudLab

End-to-end evaluation with *six* diverse workloads

- TPC-C, SEATS, Twitter, YCSB-A, YCSB-B, ResourceStresser

Multiple performance *targets*

- Max throughput, 95-th% tail latency

Different underlying configuration optimizer

- SMAC, Gaussian-Based Bayesian Optimizer (GP-BO) ; DDPG (RL-Based)

Generalization to *newer* PostgreSQL version

Ablation Studies

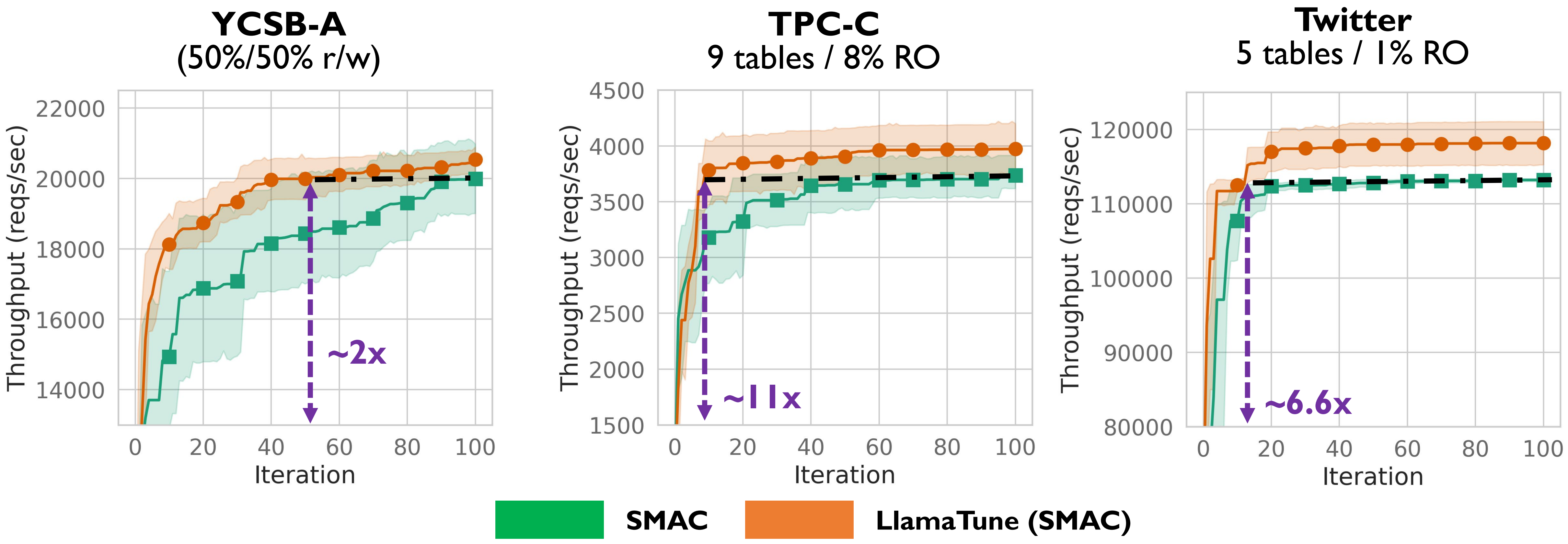
- Measure *how much* each component contributes

Sensitivity analysis for each individual component

Overhead of the configuration optimizer

End-to-End Evaluation

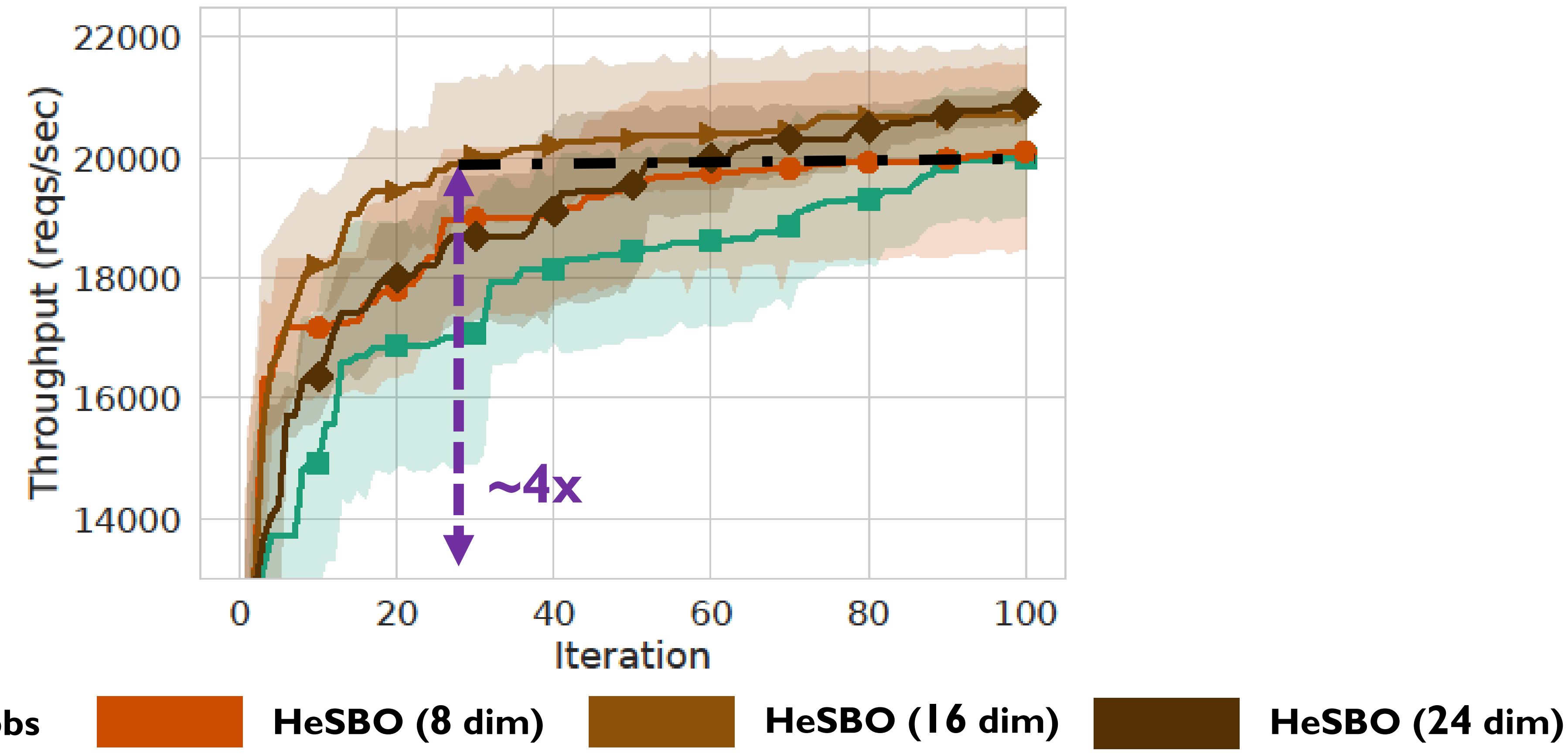
PostgreSQL v9.6, 90 knobs, SMAC, average of 5 runs



LlamaTune reaches baseline perf. **~5.6x** faster – improves final perf. **~11%**

Sensitivity Analysis – Low-Dim Tuning

PostgreSQL v9.6, 90 knobs, **YCSB-A** (50%/50% read/write), SF=20, SMAC, average of 5 runs



Conclusion

Feel free to reach out for questions!

Zero-knowledge DBMS tuners require ~100 samples to find good-performing conf.

- Sample-efficiency is *crucial*; reduces required time / resources utilization

LlamaTune: exploits domain knowledge

- Use low-dimensional projections to *indirectly* tune important knobs
- Handles *special values*, *bucketizes* knob values – search space easier to explore!

Outperforms SOTA optimizers [*SMAC*, *GP-BO*, *DDPG*]: up to 11x fewer evaluations

`github.com/uw-mad-dash/llamatune/` `kkanellis@cs.wisc.edu`

Thank you! Questions?

WISCONSIN UNIVERSITY OF WISCONSIN–MADISON Microsoft Gray Systems Lab CloudLab