Automatic Database Management System Tuning Through Large-scale Machine Learning

Reed Callahan and Yuhao Bai

Terms:

Knobs :

E.g.

The amount of memory to use for caches.

How often data is written to storage.

Metrics:

E.g.

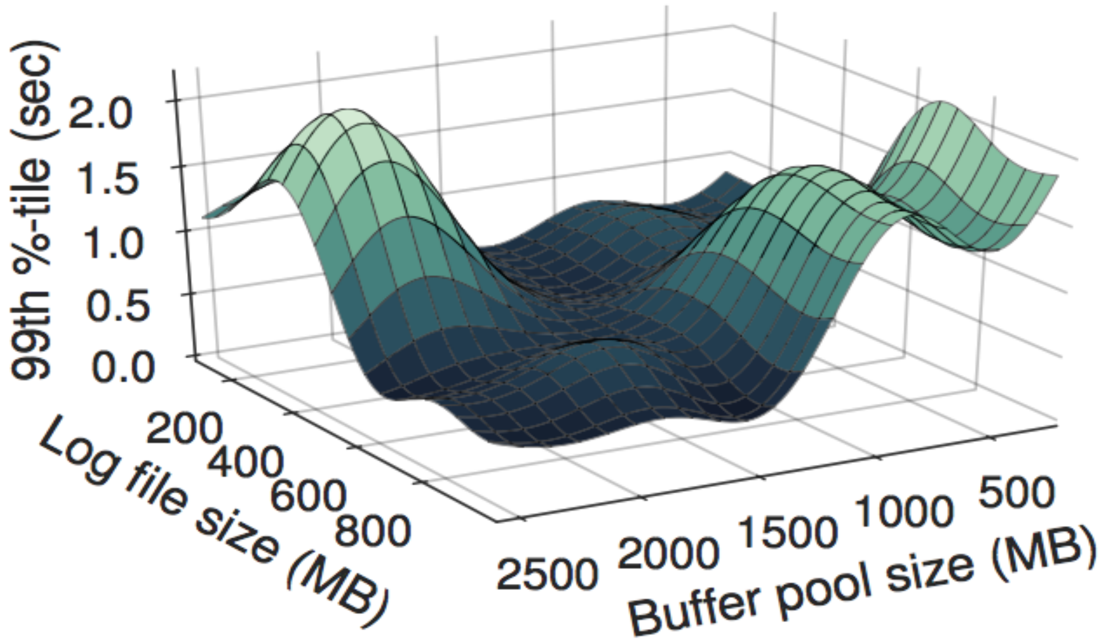
How fast DBMS can collect new data.

How fast DBMS can respond to requests.

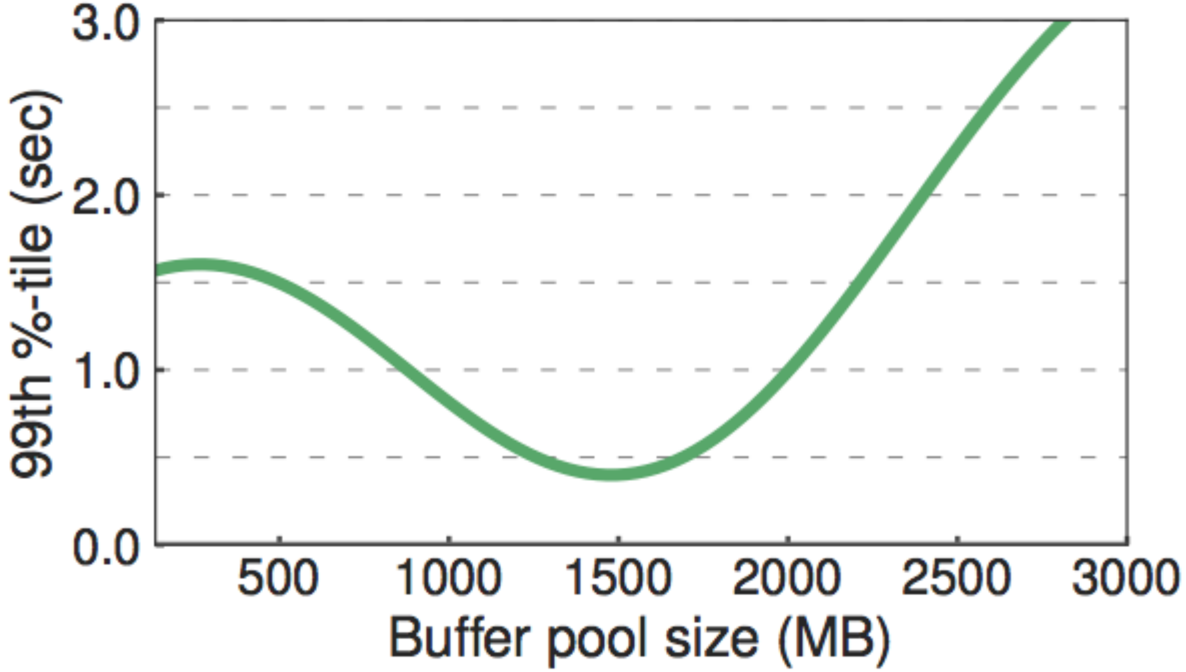
Difficulties:

- Dependencies
- Continuous Settings
- Non-Reusable Configurations
- Tuning Complexity

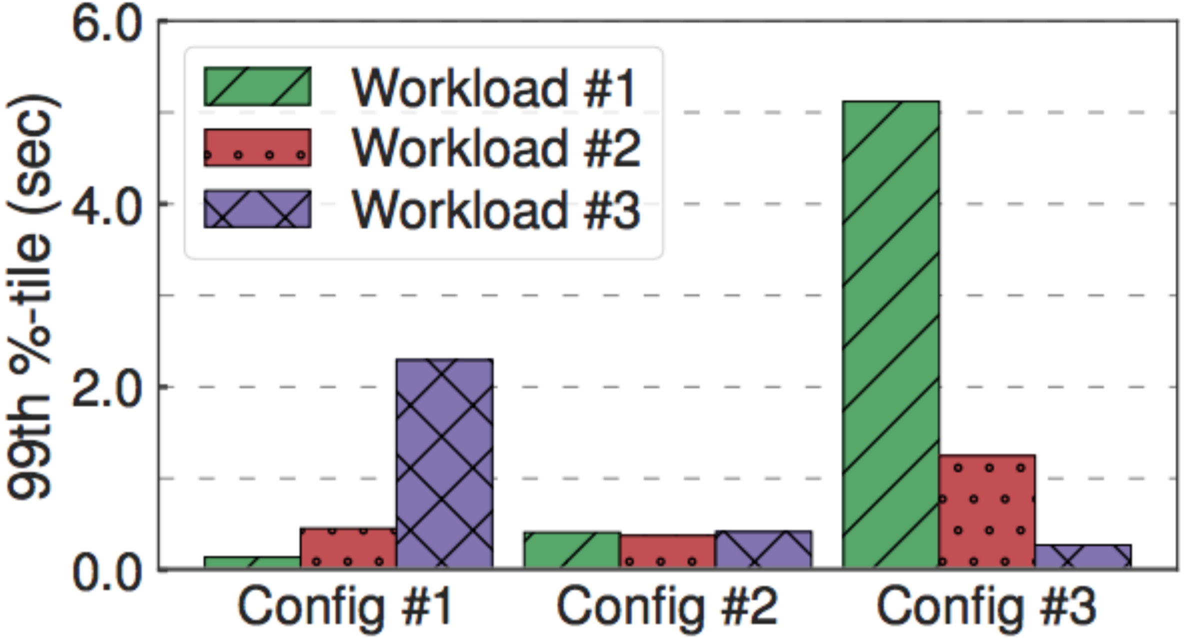
Dependencies:



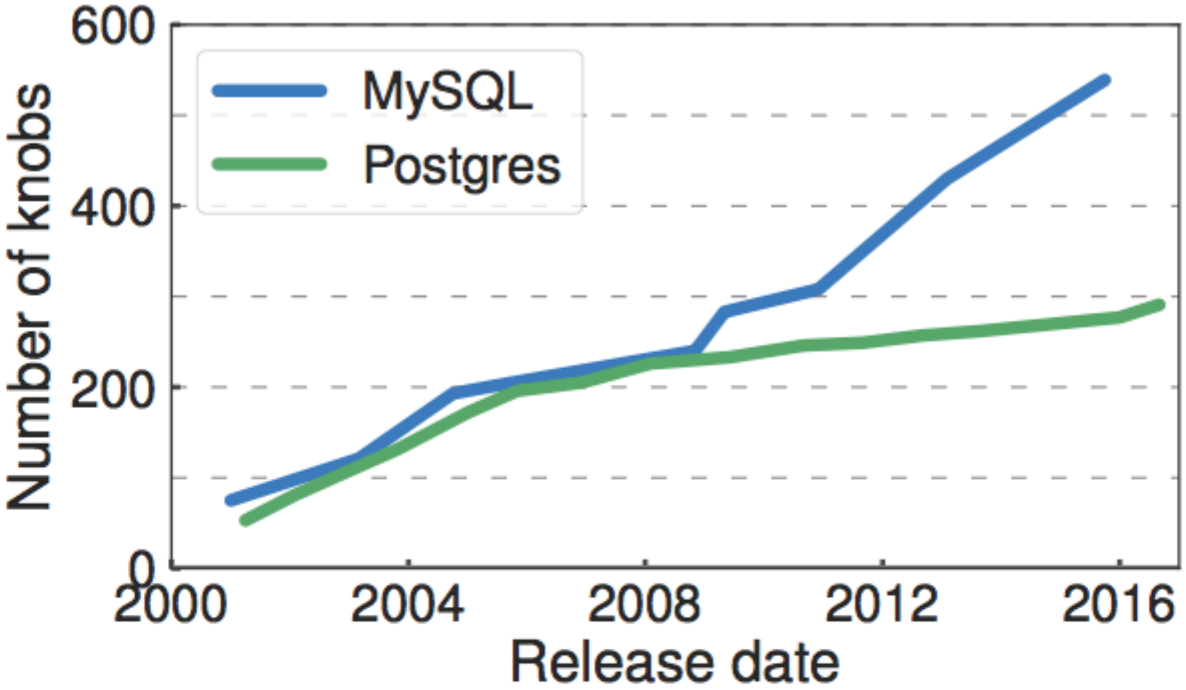
Continuous Settings



Non-Reusable Configurations



Tuning Complexity



Existing Solutions

1. Hiring experts

- Non-Reusable
- High-Cost
- Tuning Complexity

2. Existings Automatic Tuning Tools

- Dependencies
- Non-Reusable

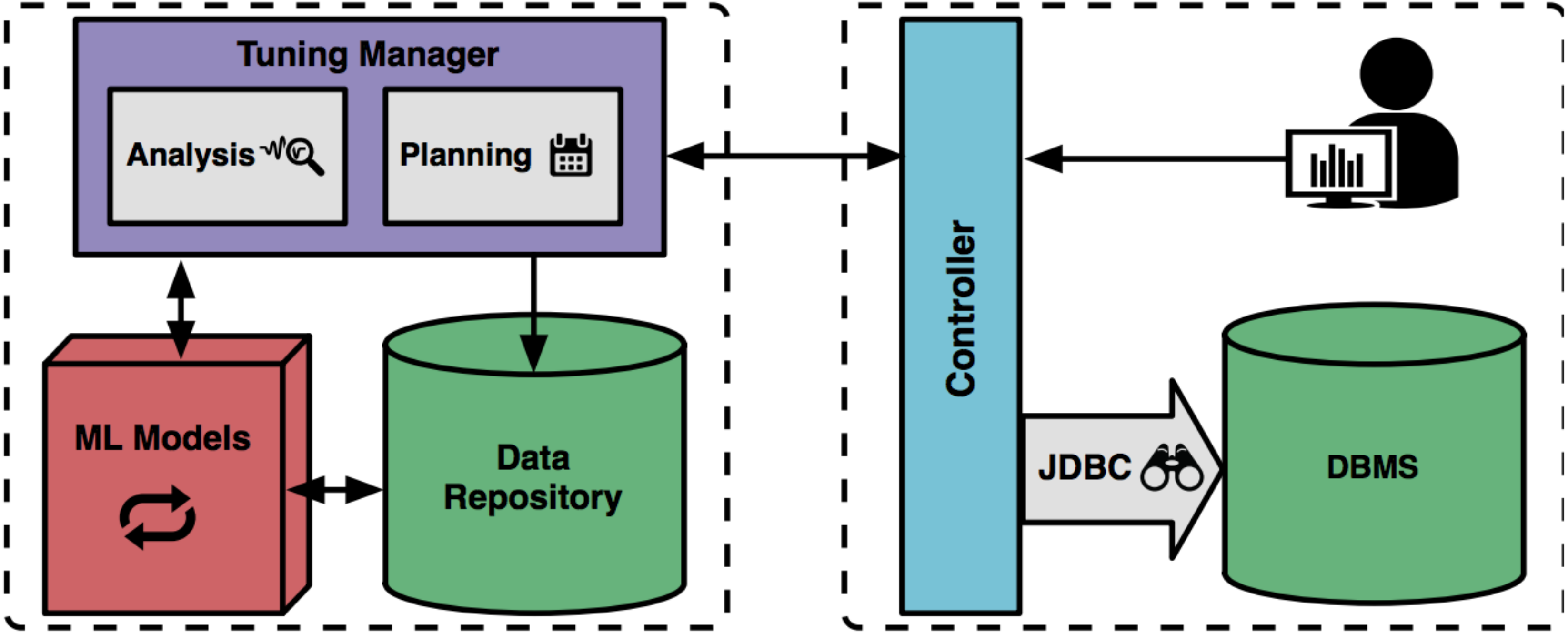
Goal:

Overcome the four problems we discussed.

OtterTune

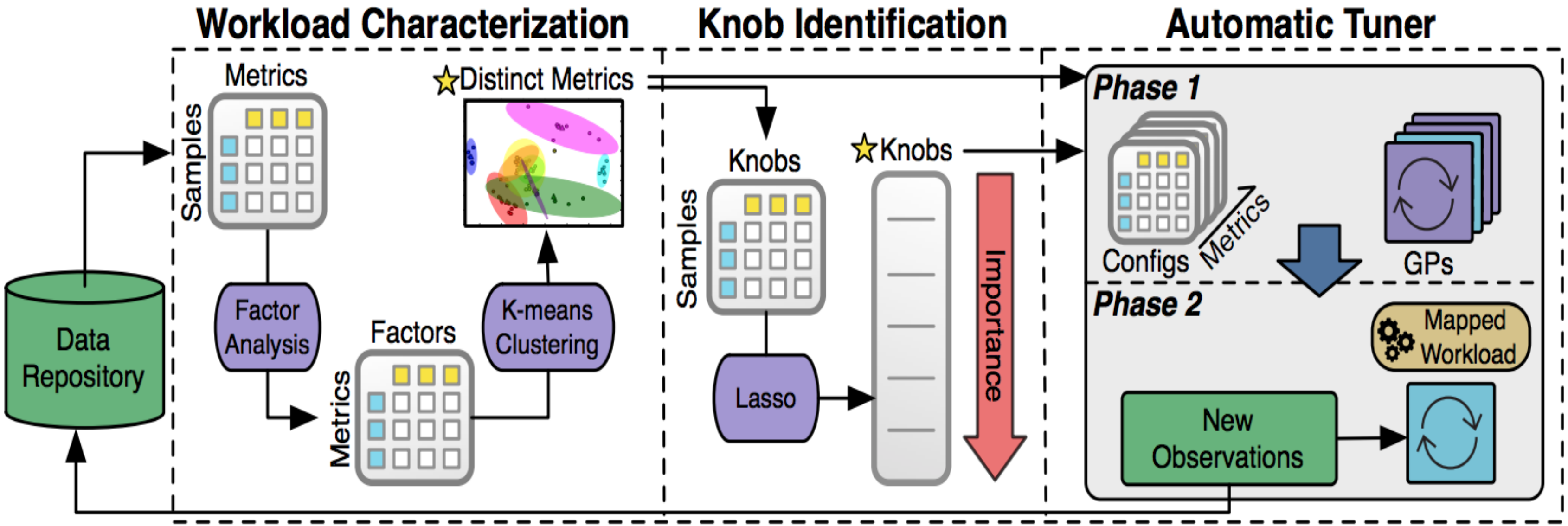
An automated approach that leverages past experience and collects new information to tune DBMS configurations.

Architecture



New Tuning Session:

1. Database Administrator (DBA) tells OtterTune what metric to optimize when selecting a configuration.
2. Controller connects to the DBMS and collects its hardware profile and knob configuration.
3. Controller then starts the first *observation period*.
4. OtterTune measures metrics chosen by the DBA.
5. Stores data in repository.
6. Modeling, Learning, and Recommendation


Workload Characterization
Knob Identification
Automatic Tuner
Data Repository
Metrics
Samples
Factor Analysis
Factors
Metrics
K-means Clustering
Distinct Metrics
Knobs
Samples
Lasso
Knobs
Importance
Phase 1
Configs
Metrics
GPs
Phase 2
Mapped Workload
New Observations

Workload Characterization

- Discover a model that best represents the distinguishing aspects of the target workload.
- Identify which previously seen workloads in the repository are similar to it.

Workload Characterization

- Statistics Collection
  - Controller.
  - DBMS's internal runtime metrics.
  - Provide a more accurate representation of a workload because they capture more aspects of its runtime behavior.
  - Metrics are directly affected by the knob's setting.
- Pruning Redundant Metrics
  - The smallest set of metrics that capture the characteristics for different workloads.
  - Speed up the process.

	Config1	Config2	Config3	......	ConfigM
Metric1					
Metric2					
Metric3					
Metric4					
Metric5					
Metric6					
Metric7					
......					
MetricN					

Redundant Metrics

- Metrics that provide different granularities for the exact same metric in the system.
  - E.g. The amount of data read in terms of bytes and pages.
- Metrics whose values are strongly correlated.

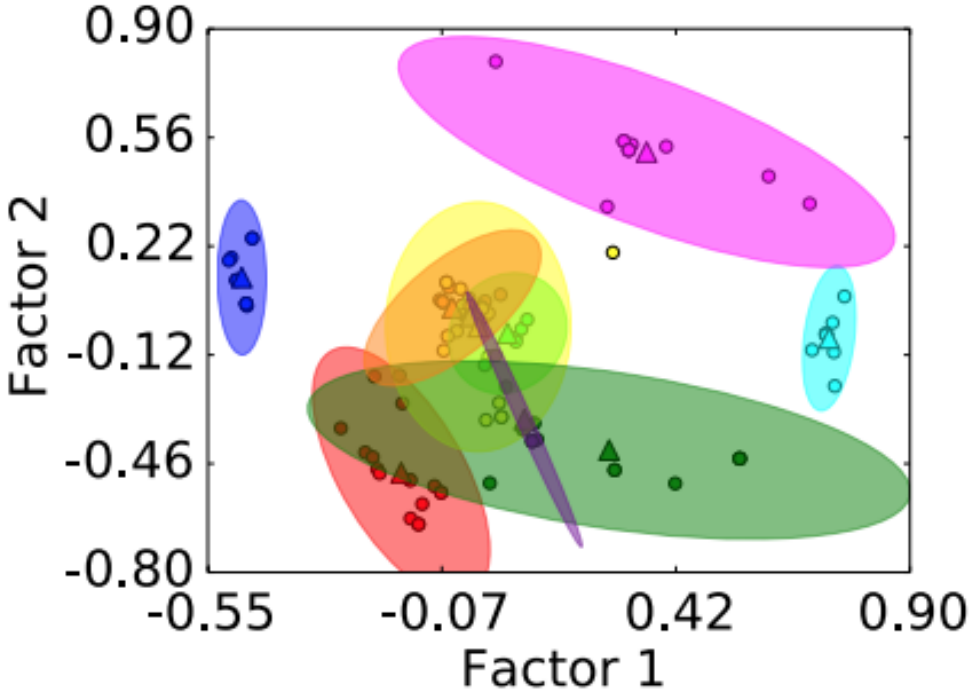
Algorithms:

1. Factor Analysis (Dimensionality Reduction): transform the high dimensional DBMS metric data into lower dimensional data.
2. K-Means (Clustering): cluster the lower dimensional data into meaningful groups.

Dimensionality Reduction (SVD Example)

$$
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{bmatrix}
=
\begin{bmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{bmatrix}
\text{ x }
\begin{bmatrix}
9.64 & 0 \\
0 & 5.29
\end{bmatrix}
\text{ x }
\begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{bmatrix}
$$

Clustering



Identifying important knobs

- After pruning the metrics, OtterTune identifies which knobs have biggest impact on DBA's target objective function
- While there can be hundreds of knobs on a DBMS, only a subset affect performance
  - Cannot reduce as this will limit configurations that should be considered
- To find positive and negative correlations between knobs and systems performance, OtterTune uses Lasso
  - Lasso is a feature selection technique for linear regression

Lasso

- Lasso uses ordinary least squares (OLS) to estimate regression weights by minimizing residual squared error
- Uses polynomial features to account for nonlinear correlations and dependencies between knobs
- Reduces effect of irrelevant variables by penalizing models with large weights
- Only keeps non-zero weight features
- Keeps only number of features based on penalty strength
- Lasso is more interpretable, stable and computationally efficient than regularization / other feature selection methods

Knob Selection

- Lasso uses ordinary least squares (OLS) to estimate regression weights by minimizing residual squared error
- OtterTune now has a ranked list of all knobs
  - Lasso path algorithm orders list of knobs by the strength of statistical evidence that they are relevant
- OtterTune decides how many knobs to use
  - Too many exponentially grows optimization time
  - Too few limits it from finding best configuration
- Automates selection through dynamically increasing the number of knobs used in a tuning session over time

Automated Tuning

- Two step analysis for each knob configuration
- Step 1: Find workload from a previous tuning session which best represents the target workload.
  - Compares current metrics against previous workloads to see determine those that will react similarly to knob settings.
  - Calculates Euclidean distance between the vector of measurements for the target workload and the corresponding vector for each workload
    - Builds score for each workload by taking average of the distances over all metrics
  - Chooses workload with lowest score (most similar)

Automated Tuning

- Step 2: chooses a configuration that is explicitly selected to maximize the target objective.
    - Path to configurations by Gaussian Process
- Gaussian Process
    - Form of regression used to recommend configurations that will improve the target metric
    - returns configuration along with the expected improvement from running this configuration to the client
    - The DBA can use the expected improvement calculation to decide whether they are satisfied with the best configuration that OtterTune has generated thus far.
- Uses Gradient Descent for initialized knob configuration

Step 2 Continued

- Tries to find a better configuration than the best current configuration. This is done by either of two options:
    1. Exploration: searching an unknown region in its GP (i.e., workloads for which it has little to no data for)
    2. Exploitation: Selecting a configuration that is near the best configuration in its GP.
        - Tries slight modifications to the knobs to see whether it can further improve the performance
- Uses gradient descent to find the local optimum on the surface predicted by the GP model using a set of configurations, called the initialization set, as starting points

Experimental Evaluation

- Performed testing on three DBMS: MySQL (v5.6), Postgres (v9.3) and Actian Vector (v4.2)
- Over 100K trials per DBMS
- All evaluations were run on Amazon EC2 with two instances
  - Instance 1: OtterTune's controller integrated with the OLTP-Bench framework.
    - Deployed on m4.large instances w/ 4 vCPUs + 16 GB RAM.
  - Instance 2: Target DBMS + tuning manager and repository
    - Target DBMS deployed on m3.xlarge instances w/ 4 vCPUs + 15 GB RAM.
    - Manager deployed on a local server with 20 cores and 128 GB RAM.

Evaluation Workloads

- YCSB - Yahoo! Cloud Serving Benchmark
  - Six OLTP transaction types that access random tuples based on a Zipfian distribution
  - Database contains a single table of 18m tuples (~18 GB) with 10 attributes
- TPC-C
  - Industry standard for evaluating performance of online transaction processing systems (OLTP)
  - Consists of five transactions with nine tables that simulate an order processing application
  - Database of 200 warehouses (~18 GB) in each experiment
- Wikipedia
  - Used for stress testing
  - The database contains 11 tables and eight different transaction types (100k articles, ~20 GB in total)
- TPC-H
  - Decision support system workload that simulates an OLAP environment with little prior knowledge of the queries
  - Contains eight tables in 3NF schema and 22 queries with varying complexity
  - Scale factor of 10 in each experiment (~10 GB)

Evaluation Results

- The optimal number of knobs for a tuning session varies per DBMS and workload

- OtterTune is able to tune DBMSs like MySQL and Postgres that have few impactful knobs, as well as DBMSs like Vector that require more knobs to be tuned

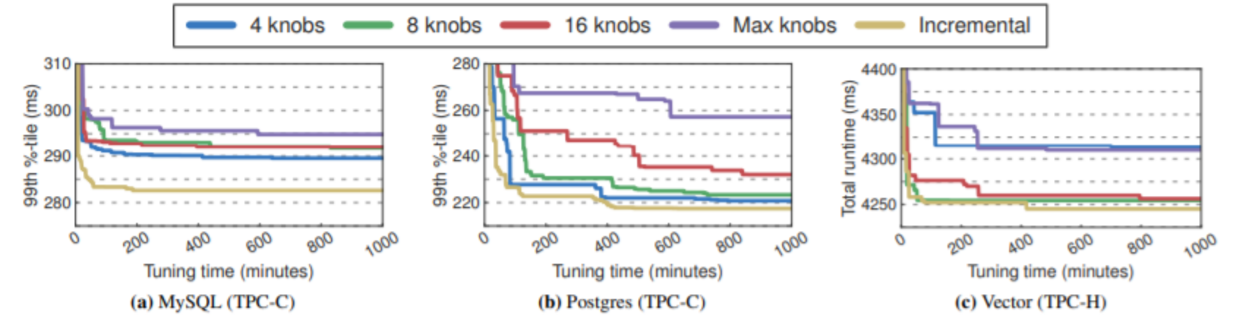
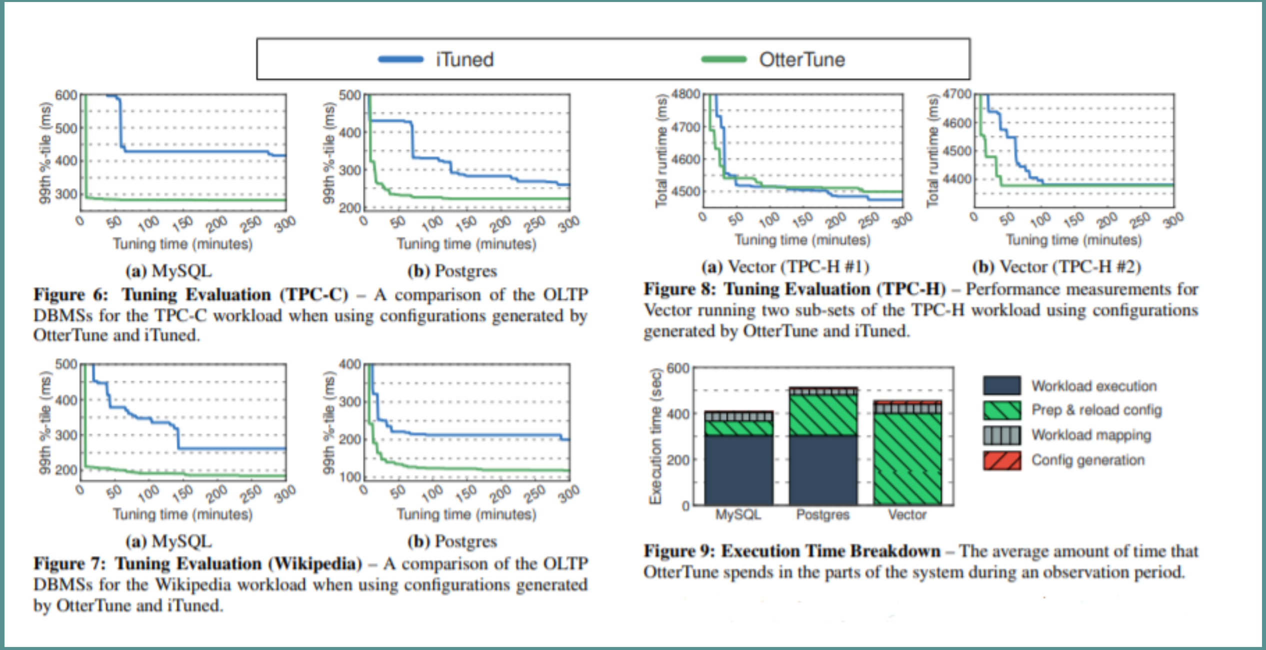


Figure 5: Number of Knobs – The performance of the DBMSs for TPC-C and TPC-H during the tuning session using different configurations generated by OtterTune that only configure a certain number of knobs.

- These results show that increasing the number of knobs that OtterTune considers over time is the best approach because it strikes the right balance between complexity and performance

Results vs. Current Solution

- Workload Execution: Time for DBMS to execute the workload in order to collect new metric data.
- Prep & Reload Config: The time that OtterTune's controller takes to install the next configuration and prepare the DBMS for the next observation period (i.e. restarting)
- Workload Mapping: Time for OtterTune's dynamic mapping scheme to identify the most similar workload
- Config Generation: Time for OtterTune's tuning manager to compute the next configuration for the target DBMS. (Gradient Descent + GP modeling)



- ITuned cannot reuse training data, so this compares performance between them

Future Work

- Approximate or generalize hardware capabilities
- Adapt techniques to optimize physical design of database

Our Thoughts + Questions?

- Possible work with ITuned to utilize their more advanced algorithms with their training data reuse process?