Brandeis
UNIVERSITY

AtlasTune

Multi-Agent Reinforcement Learning for Database Optimization

James Petullo

Brandeis University Computer Science PhD Research Retreat, May 29, 2024

Database Optimization Overview

- For optimal performance on intended workloads, a database management system (DBMS) needs to be properly configured.
- The configuration task can be broken down into four broad components:
    - Query optimizing
    - Query scheduling
    - Index selection
    - Knob tuning
- This past semester we focused on the latter two components.

Index Selection

- Choosing a subset of columns in a database's table to index is essential to improving query performance.
- Finding the optimal subset is challenging, as the selector must balance tradeoffs between increasing the speed of read operations, memory usage, and the impact the chosen indexes have on write operations.
- Automatic index selection tools range from simple heuristics to reinforcement learning.
- We have implemented a custom Deep Q Learning (DQN) agent for index selection, to be used alongside our knob tuner in our multi-agent optimizer.

Knob Tuning

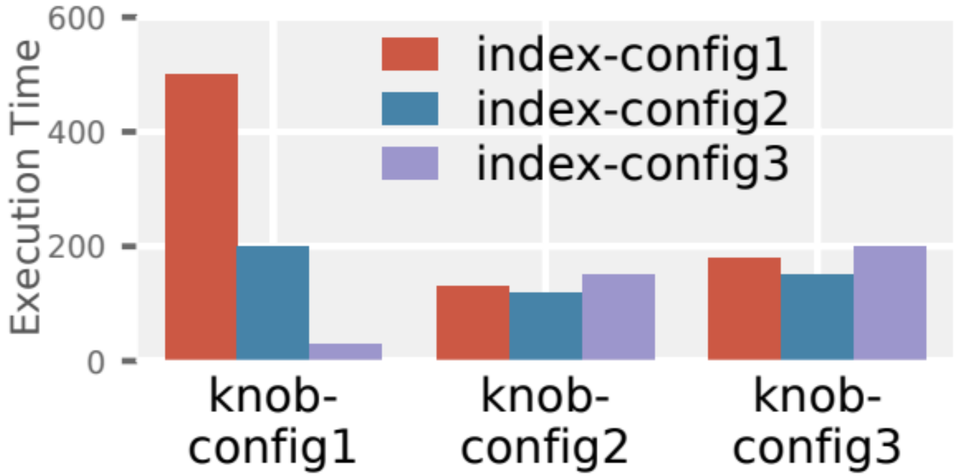
- DBMSs have hundreds of configuration parameters (PostgreSQL has over 350 knobs).
- Proper configuration values are vital for database performance, particularly on large systems with complex workloads.
- Knobs have many intricate interdependences in their own right, and it is non-trivial to tune by hand, especially considering the fact that the range of values many knobs can be set to are very large (1 to 100000, in some cases).
- We implemented a deep deterministic policy gradient (DDPG) agent to recommend optimal knob configurations across continuous action space.

Component Interdependencies

- The standard optimization approach is to run individual, custom ML-based tuners on each component sequentially.
- Unfortunately, this process does not take into account the interdependencies that exist between components, whereby the optimal configuration of one component depends on the configuration of the other components.
    - For example, a large query cache and small buffer size (knob tuner) is best when no indexes are built (index selection) while a smaller cache size and larger buffer pool are more suitable when indexes are built.
- If these interdependencies are not accounted for, tuners will ultimately suggest suboptimal configurations that can degrade overall DBMS performance

Component Interdependencies, cont.

- Example of workload execution times under different knob and index configurations. Index-config3 and knob-config1 together perform best, as opposed to the others.



(b) Dependency Between DBMS Components.

Previous Work and Existing Solutions

- The literature is very sparse although one paper in particular, UniTune, outlines a rudimentary component communication protocol.

- UniTune utilizes out of the box, pre-existing ML-based tuners, runs them sequentially to convergence, and attempts to achieve component communication and coordination by sharing states. There are some problems, however:

    - Agents do not possess reward functions that account for the reward signals of the other agents, and each utilize different workloads and metrics.

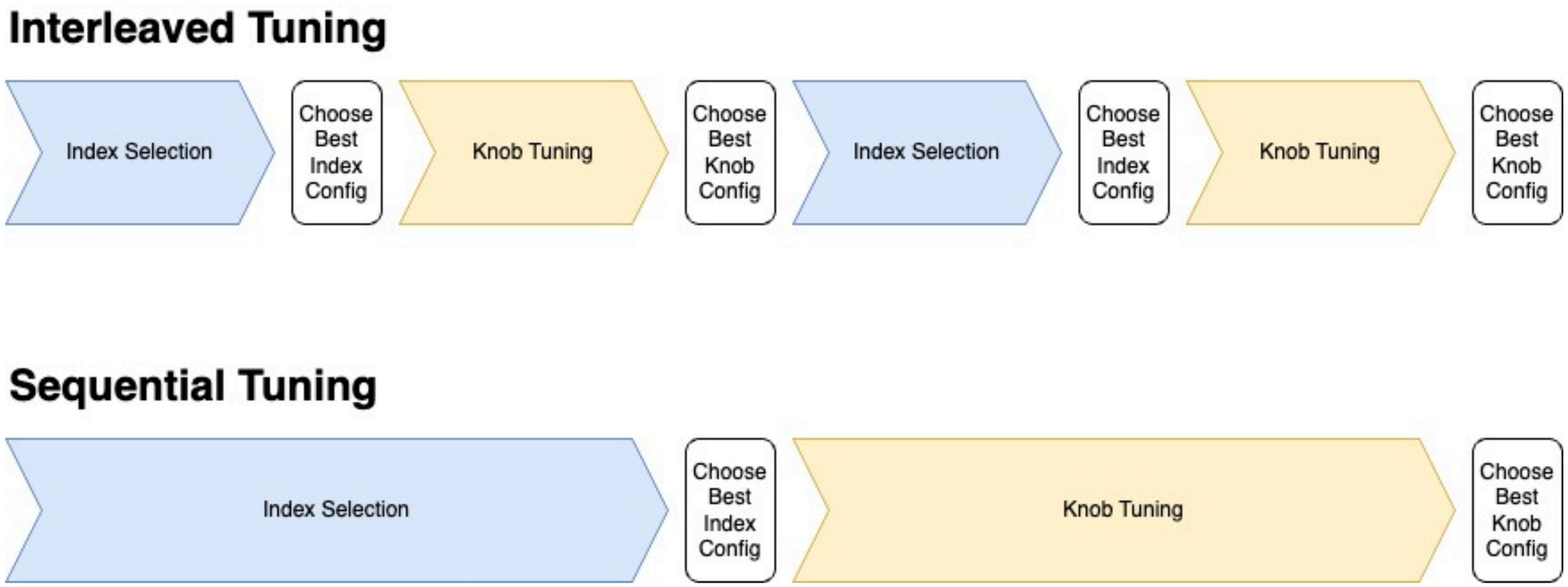
    - Components such as query scheduling and query optimizing are tuned at the same time as knobs and indexes, offline.

Our Solution

- State sharing: each tuning component's state includes not just its own default state but that of the other agent.
- Team reward: both the knob tuner and the index selector have the same reward function, which uses the workload cost of a set of queries as a reward signal.
- Interleaved tuning: instead of running one agent to convergence and then deploying the next agent, we alternate between tuners, pausing the execution of one after a certain number of iterations, fixing the best configuration found, and then switching to the next agent. This way, each agent has the chance to adapt to into intermediate decisions and compensate for any suboptimal solutions the other makes.

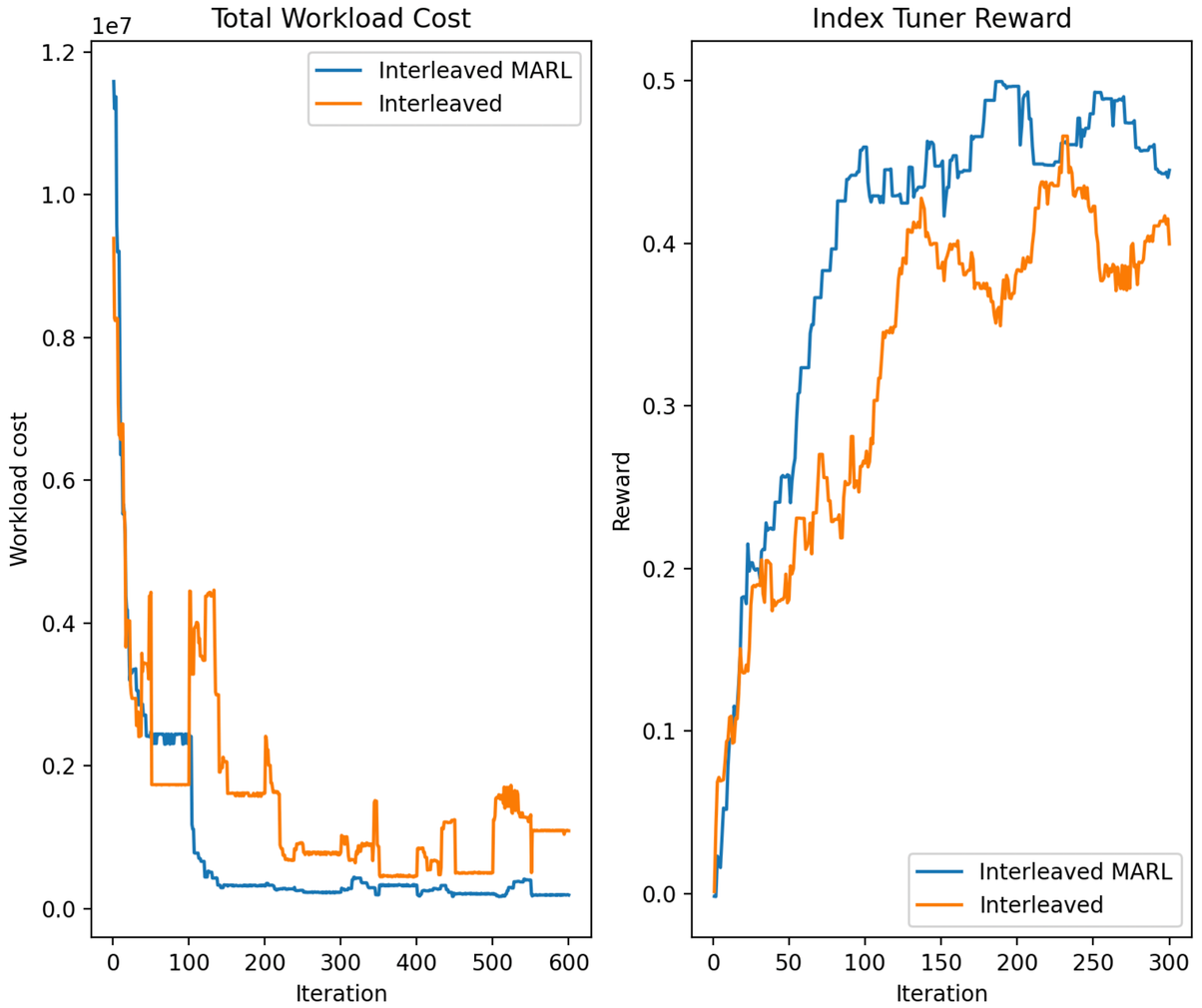
Our Solution, cont.

- Interleaved tuning: at successive intervals, an agent's best configuration is fixed and tuning switches to the next agent:



Preliminary Results

- Using a workload of twelve read-only queries on a 4 GB database:



Future Work

- We are setting up a new benchmark on an even larger database with a more complex workload to gauge performance.
- Integrate an existing MARL query scheduler and query optimizer with AtlasTune’s knob tuner and index selector.

References

- Xinyi Zhang, Zhuo Chang, Hong Wu, Yang Li, Jia Chen, Jian Tan, Feifei Li, Bin Cui: “A Unified and Efficient Coordinating Framework for Autonomous DBMS Tuning”, 2023; arXiv:2303.05710

- Chi Zhang, Olga Papaemmanouil, Josiah P. Hanna, Aditya Akella: “Multi-agent Databases via Independent Learning”, 2022, AIDB@VLDB 2022 Proceedings of 4th International Workshop on Applied AI for Database Systems and Applications; arXiv:2205.14323