Rethinking Database Tuning By Learning To Split

ANDY HUYNH, Boston University, USA

Tuning Data Systems. Tuning data systems is a notoriously hard problem [2, 3, 10], and continues to grow in complexity as databases further expand their functionality [5, 7, 8]. The tuning process typically is composed of three parts: an *objective* to optimize, either in the form of a cost model or from experimental metrics; a set of *tuning knobs* or configuration; and *expected input knowledge* such as the expected workload and execution environment. Together, they form the basic components required to formulate a database tuning problem where the goal is to find a *configuration* that optimizes an *objective*, based on *expected knowledge* (e.g. workload, available resources) [1, 4, 9].

Learning to Split the Tuning Problem. When tuning, database designers are effectively solving the problem—*What is the optimal configuration for a particular workload?* However, to come up with an answer, we are required to answer an auxiliary question—*What is the cost of a workload executed on a given configuration?* Traditional auto-tuning methods focus on the former question and implicitly solve the latter, however, solving these in tandem inherently creates a more difficult problem to solve than if we were to tackle them separately. Therefore, we propose to rethink our approach to tuning to instead split the problem into separate tasks that follow naturally from these questions; learn the cost model, and then learn to navigate this cost to find an optimal.

Leveraging Learning. While modern databases may already have cost models associated with them, they rarely exhibit *nice* properties such as smoothness, continuity, and convexity; this creates restrictions on the types methods we can use to navigate the cost surface. Therefore, we utilize a neural network as a surrogate for our cost model—a *learned cost model.* This surrogate guarantees that the tuning objective exhibits nice properties; therefore allowing us to utilize modern optimization techniques to optimally navigate the cost surface. Additionally, we may design networks to handle esoteric and special settings, (e.g., buffer_size = -1) or categorical knobs that generally pose challenges in the classic tuning setting.

Keeping Your Expert. The largest challenge of using a learned cost model is—*How do we reasonably generate data points to learn from?* While deploying a database system and executing a workload can be costly, DBAs already perform this learning step offline through simulations or playing traces. Additionally, a plethora of modern databases already utilize cost models to optimize their configurations, and while they may accurately estimate database performance, they fail to exhibit nice properties that enables modern optimization techniques. However, we can *keep the expert* and utilize analytical cost models to augment datasets to train learned cost models, for example we are currently working on LSM trees and the variety of cost models available in literature.

Unlocking Augmented Tuning With Unexpected Inputs. When deploying database systems, we rarely encounter cases where the executed workload precisely matches our expected workload. We recently explored this concept and how to accommodate this uncertainty in tuning via robust tuning in a paradigm called ENDURE [6]. While we successfully demonstrated that robust tunings provide performance benefits under uncertainty, the paradigm falls short of being easily generalizable to other systems. However, with a split tuning paradigm, accommodating uncertainty becomes trivial. We simply train a separate model to robustly navigate our cost, and better yet, we are working on relaxing the robustness constraint and optimize for arbitrary latency percentiles such as p99 latency, a more practical real-world metric. Learned models enable this by being continuous and differentiable, therefore allowing us to utilize modern optimization techniques to navigate the cost surface. Overall, by splitting the tuning process, we will be able to solve new variants of tuning problems that were infeasible beforehand either through classic optimizers or learned tuners.

Author's address: Andy Huynh, ndhuynh@bu.edu, Boston University, USA.

REFERENCES

[1] Dana Van Aken, Andrew Pavlo, Geoffrey J Gordon, and Bohan Zhang. 2017. Automatic Database Management System Tuning Through Large-scale Machine Learning. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 1009–1024. https://doi.org/10.1145/3035918.3064029

[2] Surajit Chaudhuri, Benoit Dageville, and Guy M Lohman. 2004. Self-Managing Technology in Database Management Systems. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*. 1243. https://doi.org/10.1016/B978-012088469-8.50116-9

[3] Surajit Chaudhuri and Gerhard Weikum. 2005. Foundations of automated database tuning. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 964–965. https://doi.org/10.1145/1066157.1066305

[4] Biplob K. Debnath, David J. Lilja, and Mohamed F. Mokbel. 2008. SARD: A statistical approach for ranking database tuning parameters. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*. 11–18. https://doi.org/10.1109/ICDEW.2008.4498279

[5] Marc Holze, Ali Haschimi, and Norbert Ritter. 2010. Towards workload-aware self-management: Predicting significant workload shifts. *Proceedings of the IEEE International Conference on Data Engineering (ICDE)* (2010), 111–116. https://doi.org/10.1109/ICDEW.2010.5452738

[6] Andy Huynh, Harshal A. Chaudhari, Evimaria Terzi, and Manos Athanassoulis. 2022. Endure: A Robust Tuning Paradigm for LSM Trees Under Workload Uncertainty. *Proceedings of the VLDB Endowment* 15, 8 (2022), 1605–1618.

[7] C Mohan. 2016. Hybrid Transaction and Analytics Processing (HTAP): State of the Art. In *Proceedings of the International Workshop on Business Intelligence for the Real-Time Enterprise (BIRTE)*.

[8] Fatma Özcan, Yuanyuan Tian, and Pinar Tözün. 2017. Hybrid Transactional/Analytical Processing: A Survey. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 1771–1775. https://doi.org/10.1145/3035918.3054784

[9] Karl Schnaitter, Serge Abiteboul, Tova Milo, and Neoklis Polyzotis. 2006. COLT: Continuous On-Line Database Tuning. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 793–795. https://doi.org/10.1145/1142473.1142592

[10] Dennis E Shasha and Philippe Bonnet. 2002. Database Tuning: Principles, Experiments, and Troubleshooting Techniques. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*.