# Bao: Learning to Steer Query Optimizers

Ryan Marcus, Parimarjan Negi, Hongzi Mao, Nesime Tatbul,
Mohammad Alizadeh, and Tim Kraska

Presentation by George Barbulescu | R244 | 23/11/2022

# Problem Statement



- What is a database query?
  - A user request to a database system
  - An act of data manipulation or retrieval
- What is a Query Optimiser?
  - A component that sits in the *Database Management System (DBMS)*
  - Receives a user request and returns a query plan
  - The goal: the query plan should be efficient (e.g., low latency)
  - Question: Can we improve the Query Optimizer with ML? How?

# Query Optimisation Example

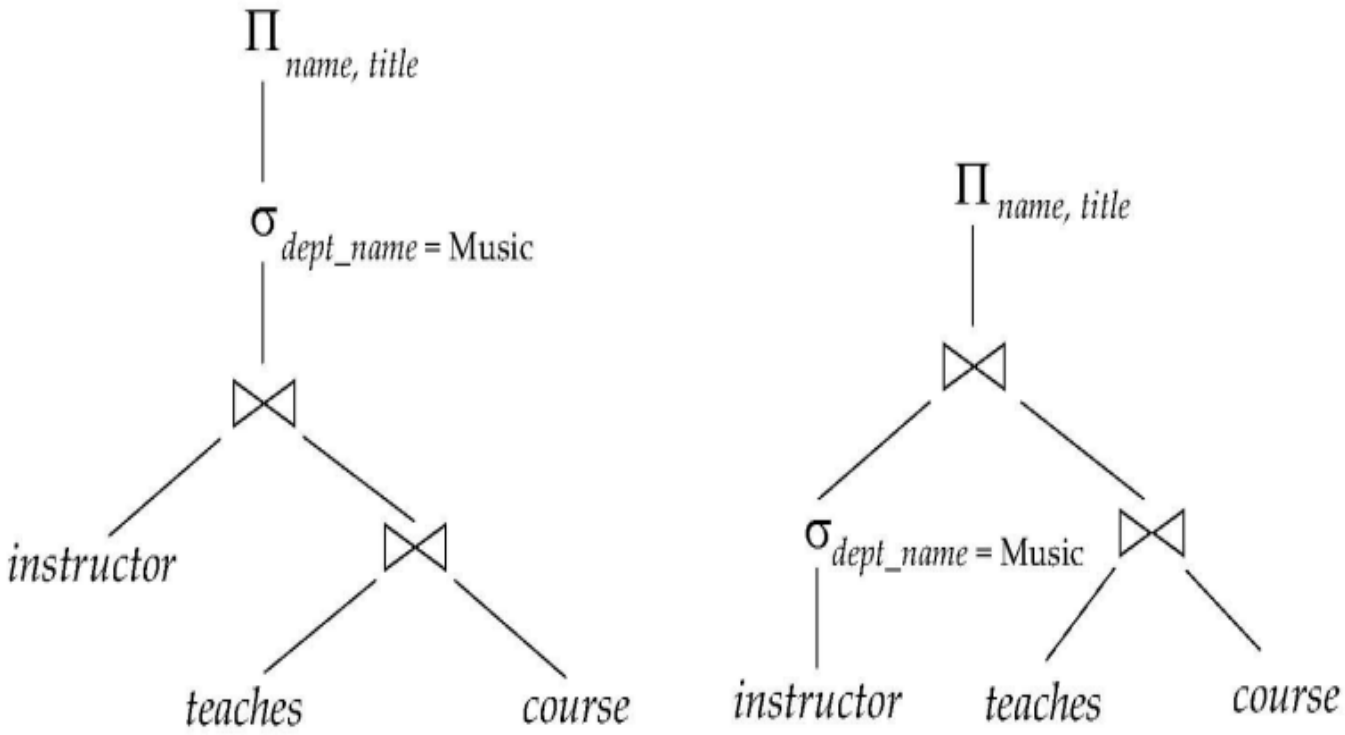
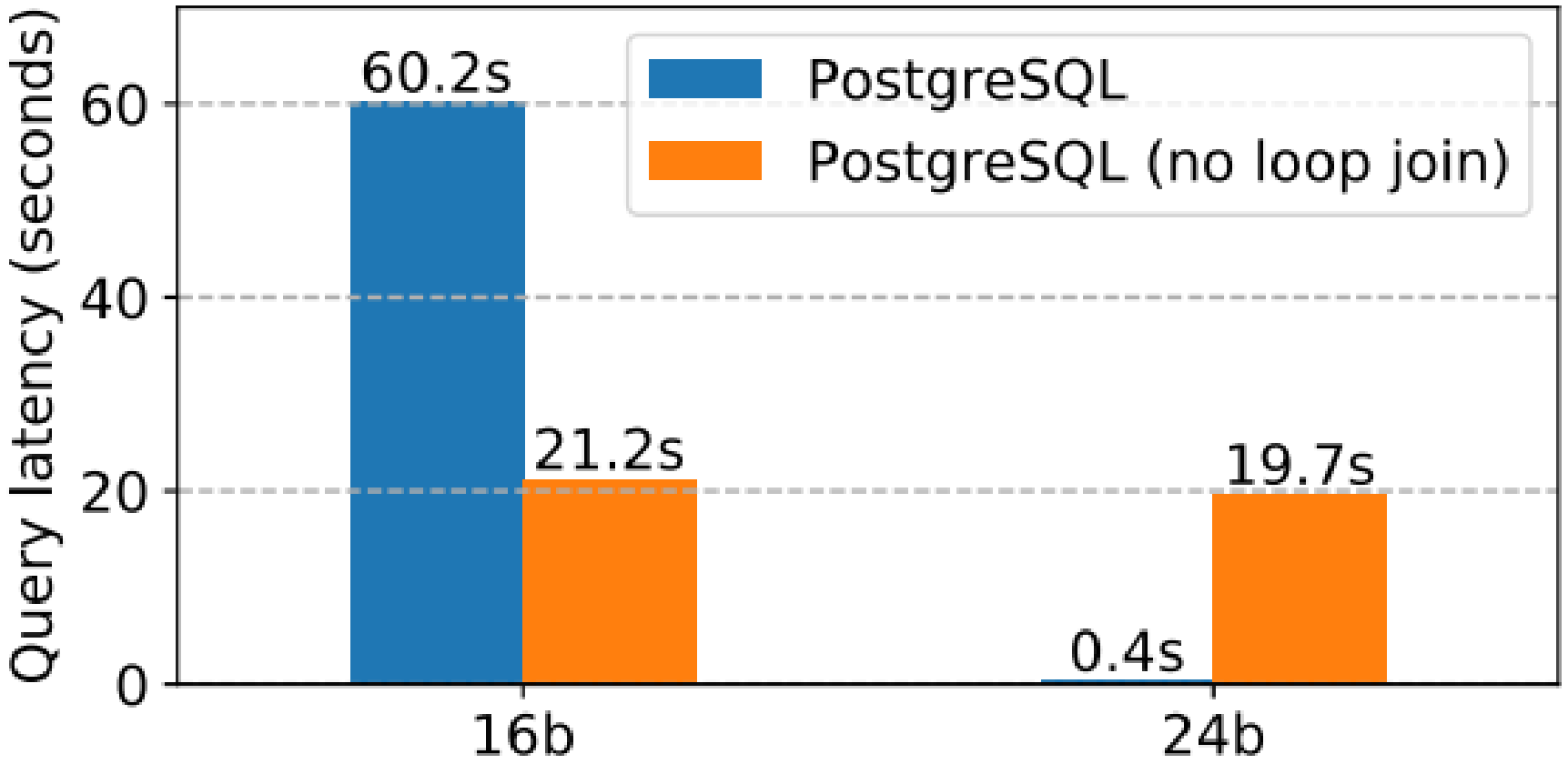



Figure taken from http://www.cbcb.umd.edu/confcour/Spring2014/CMSC424/query_optimization.pdf (University of Maryland)

# Bao
## Motivation & Background

- The Query Optimiser in DBMSs does a good job! Can we do better?
- Previous attempts replaced the Query Optimiser (Neo, QPPNet)
- Dismisses decades of white-box expertise; hard to train; meagre 99$^{th}$ percentile latency, incompatible with new schema/datasets.
- Idea: Embed DBA expertise on top of a vanilla Query Optimiser
- DBAs can reason about classes of queries; generalisation may lead to regression
- Problem: find the rules that work best given a specific query




Unless otherwise noted, figures are taken from Marcus et al., 2020
(Bao: Learning to Steer Query Optimizers )

# Bao
## Bird's eye view

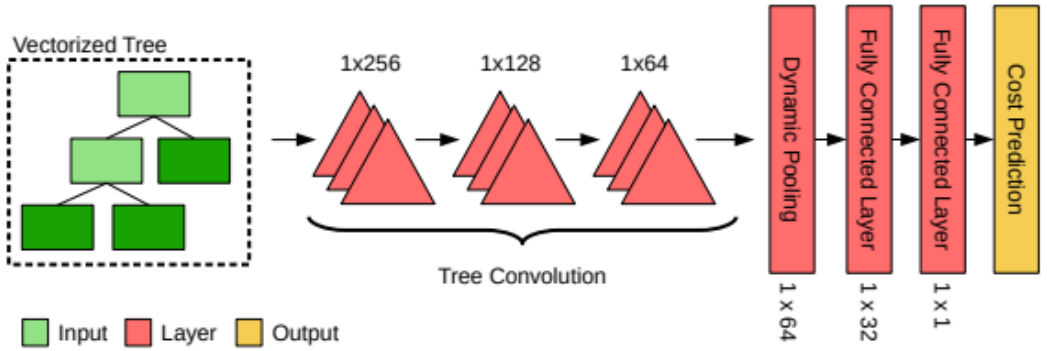
- Contextual Multi-Armed Bandit in Query Optimisation
- Context: "n" query plans generated by the set of hint sets
- Arms: a set of "hint" sets
- What is the is simplest hint set?
- Empty set! [ ]
- How about a good example? [disable index scan, disable merge join]

- Last ingredient: Regret minimisation problem

$$Regret = (Cost(Select(plan_h)) - min_i(Cost(plan_{H_i})))^2$$

# Tree Convolutional Neural Network

- How to model the performance of a query plan?
- Inductive Bias in Query Optimisation?
- Marcus et al. proposed a solution in Neo
- TCNN (Tree Convolutional Neural Networks)



# Thompson Sampling for NN



- Exploration vs Exploitation
- Prevent selecting the same hint set; Explore new hint sets for a given query
- Machine Learning aims to find the weights that fit the data best; Recall $\max(P[w|\ data])$
- New task: $(P[w|\ data])$
- Train a Neural Network on a "bootstrap" of the training data (Ian Osband et al. 2015)
- Not a new concept! Bagging – train members of the model on different bootstrap samples with replacement

# Bootstrap training set for NN
## Bagging Example

- w0 ~ $P([w \mid data])$
- Equivalent to drawing a sample from the probability distribution of weights
- Bao: retrain periodically with bootstrap samples

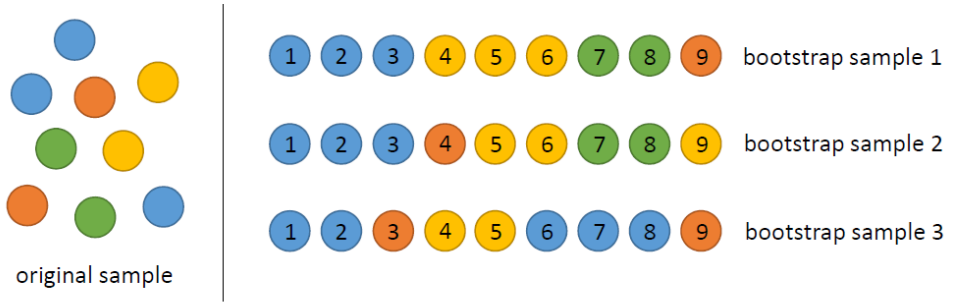
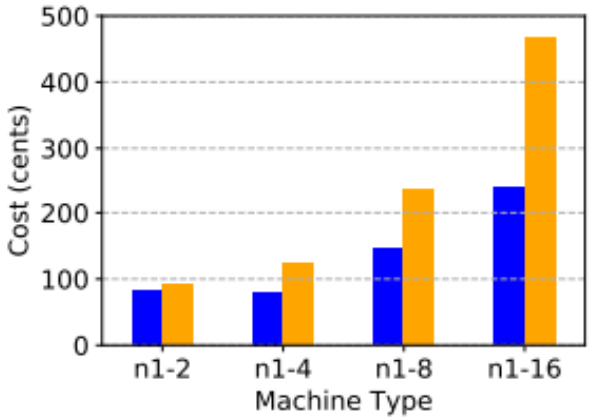
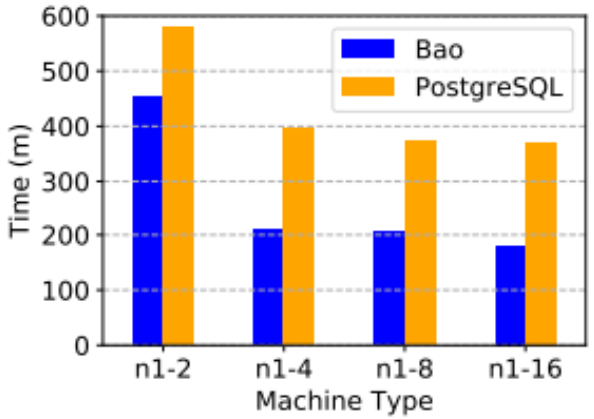
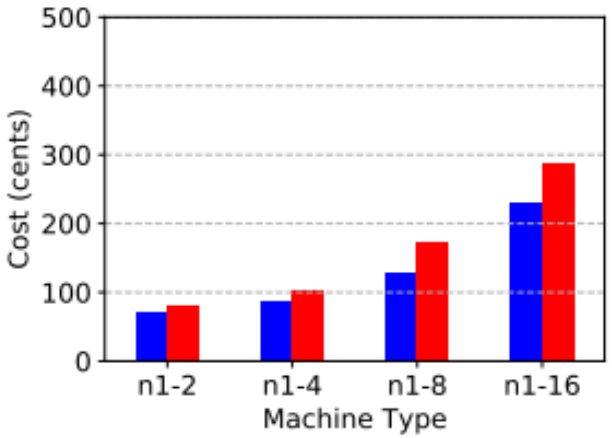


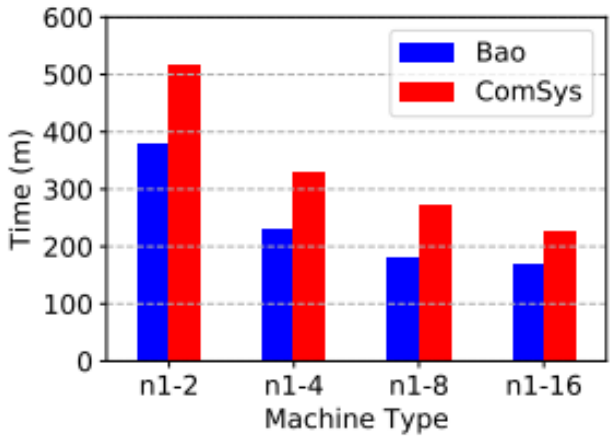

Figure taken from (Paola Galdi et al., 2018)

# Evaluation


500
400
300
200
100
0
Cost (cents)
n1-2
n1-4
n1-8
n1-16
Machine Type

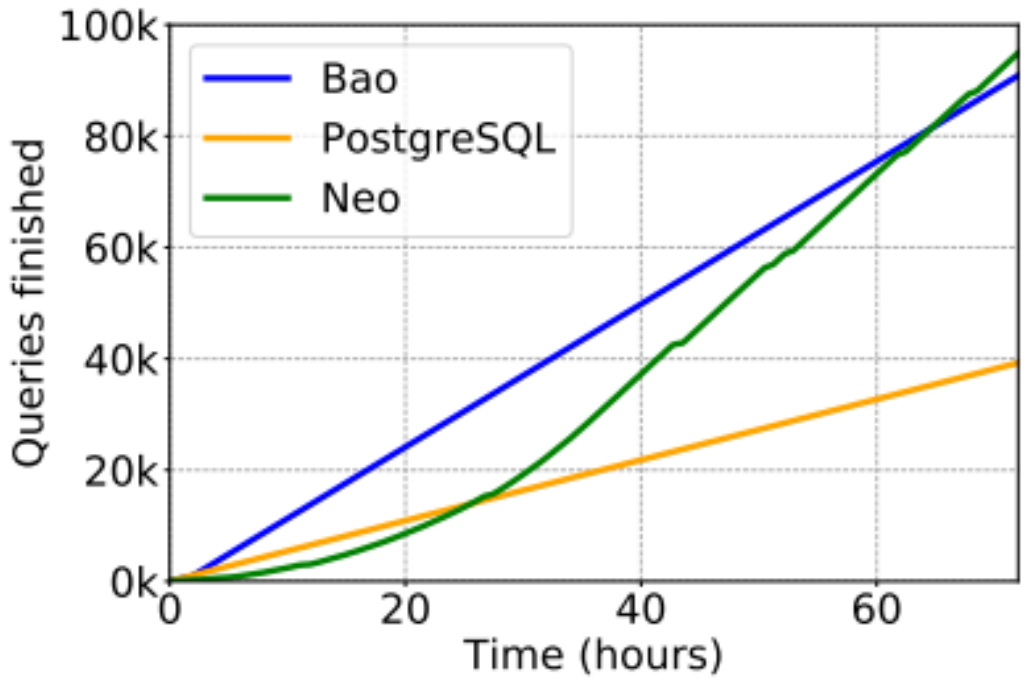

600
500
400
300
200
100
0
Time (m)
Bao
PostgreSQL
n1-2
n1-4
n1-8
n1-16
Machine Type

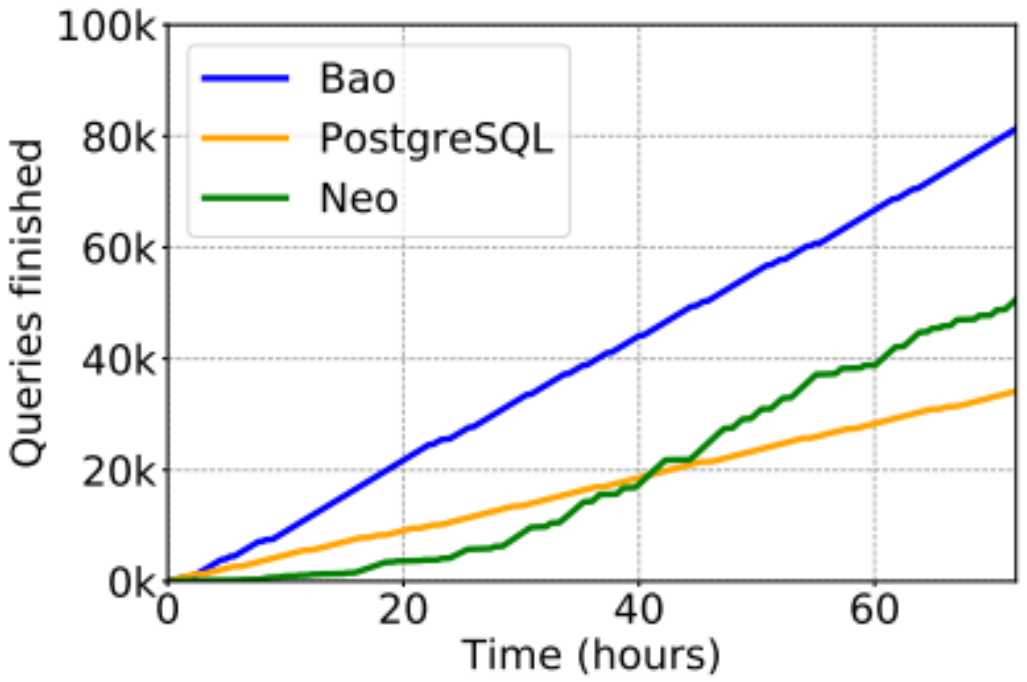

500
400
300
200
100
0
Cost (cents)
n1-2
n1-4
n1-8
n1-16
Machine Type


600
500
400
300
200
100
0
Time (m)
Bao
ComSys
n1-2
n1-4
n1-8
n1-16
Machine Type

# Learnt Engine comparison

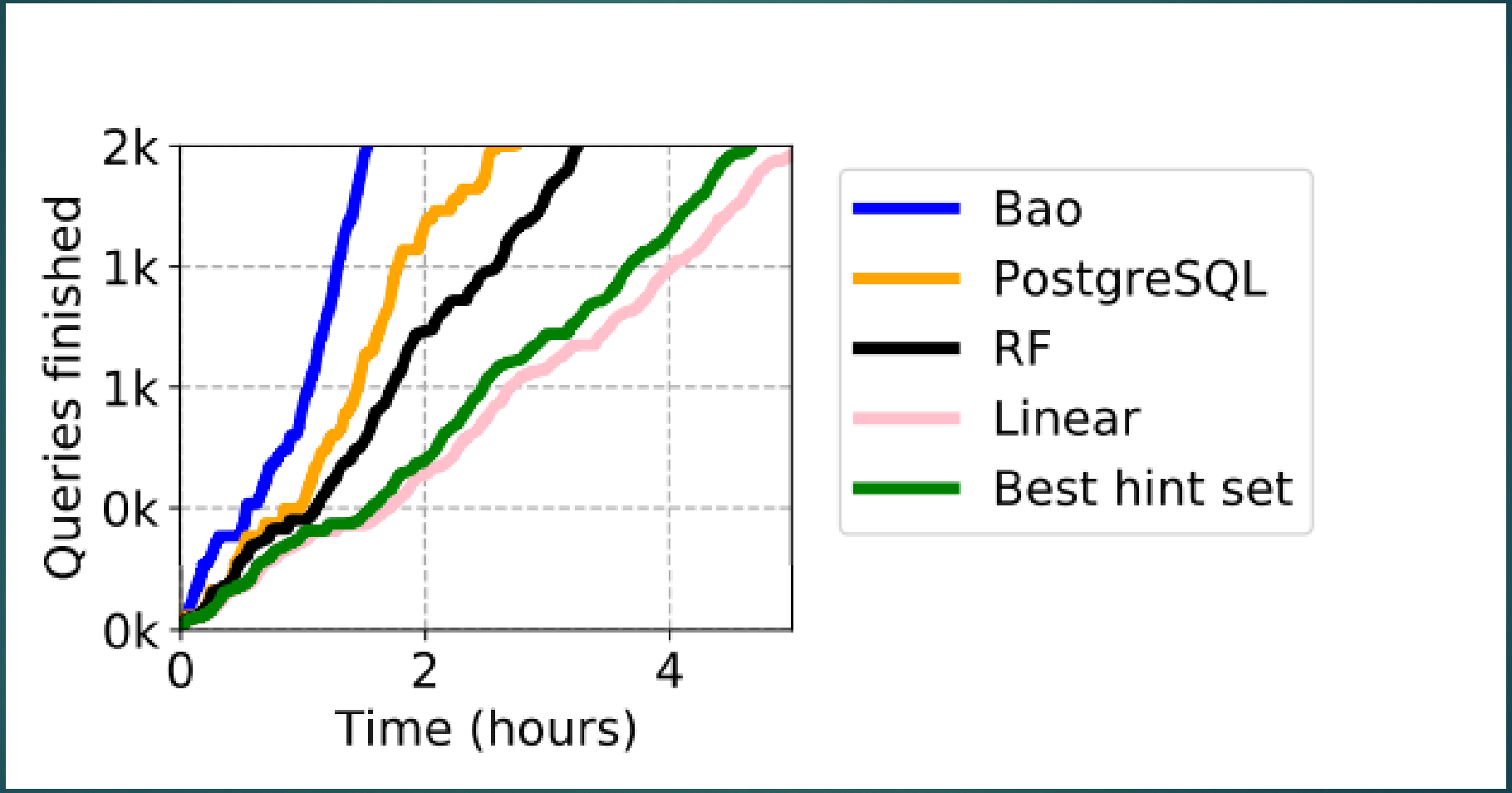




# Architecture Evaluation





# Critique Strengths

- A robust example of Machine Programming
- Unique strategy to embed white-box knowledge on a per-query basis
- Reduced training time compared to deep RL approaches
- Can be easily integrated into existing DBMSs (given expertise!)
- Able to generalise to schema and dataset changes.

# Critique
## Weaknesses & Future work

- DBA expertise is paramount to Bao's performance
- The number of arms increases exponentially with new operators
- There's no experimental evaluation or knowledge around how to find a "good" set of hint sets
  - Hints (database hooks) are not DBMS agnostic
  - A hint set should be valid (invalid: disable all join types)
- Future work:
  - Explore strategies for automated hint set generation and validation
  - Integrate into a self-driving DBMS (NoisePage [https://noise.page/])

# Thank you!

# References




- Marcus, Ryan, et al. "Bao: Learning to steer query optimizers." arXiv preprint arXiv:2004.03814 (2020).
- https://www.youtube.com/watch?v=nEy90-WNkjo&ab_channel=RyanMarcus
- Marcus, Ryan, et al. "Bao: Making learned query optimization practical." ACM SIGMOD Record 51.1 (2022): 6-13.
- Marcus, Ryan, and Olga Papaemmanouil. "Plan-structured deep neural network models for query performance prediction." arXiv preprint arXiv:1902.00132 (2019).
- https://noise.page/

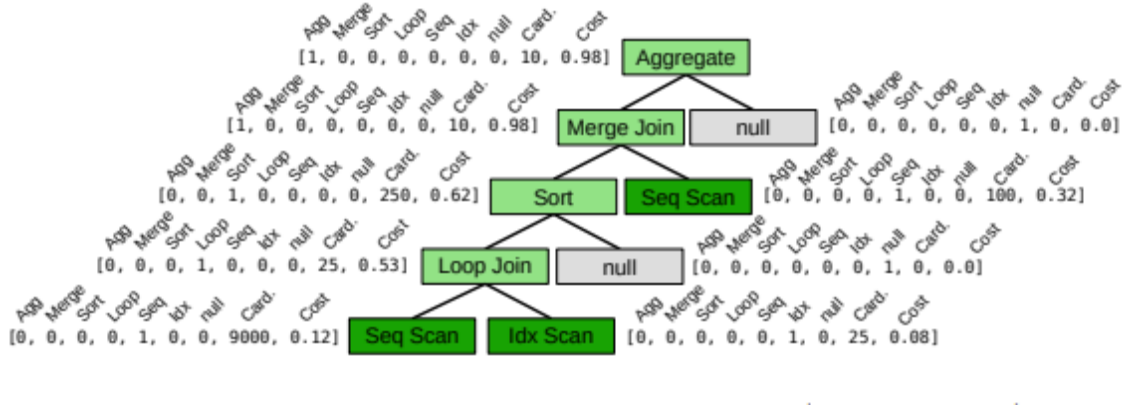
# References




- Galdi, Paola, and Roberto Tagliaferri. "Data mining: accuracy and error measures for classification and prediction." Encyclopedia of Bioinformatics and Computational Biology (2018): 431-436.
- Osband, Ian, and Benjamin Van Roy. "Bootstrapped thompson sampling and deep exploration." arXiv preprint arXiv:1507.00300 (2015).
- Breiman, Leo. "Bagging predictors." Machine learning 24.2 (1996): 123-140.
- R. Marcus, P. Negi, H. Mao, C. Zhang, M. Alizadeh, T. Kraska, O. Papaemmanouil, and N. Tatbul. Neo: A Learned Query Optimizer. PVLDB, 12(11):1705–1718, 2019.

# Appendix 1: TCNN input

- Binary query plans as input to TCNN
- Key ideas: append null nodes; split the tree if more than 2 children
- One-hot encoding for physical operators
- Agnostic of DBMS schema



# Appendix 2: Continued
## Bird's eye view



- Thompson Sampling to solve CMAB
- Not a new contribution; Machine Programming!
- Train a predictive model $M_w$ for the plan cost;
- Classic ML training: $\max(P[w|\, data])\ \ (Exploitation)$
- Guessing $w0 \sim P(w)\ \ (Exploration)$
- Balance $w0 \sim P(w|data)$