AutoSteer: Learned Query Optimization for Any SQL Database
Christoph Anneser†  Nesime Tatbul‡  David Cohen§  Zhenggang Xu∞
Prithviraj Pandian∞  Nikolay Laptev∞  Ryan Marcus∞
†Technical University of Munich  §Intel  ‡MIT  ∞Meta  ∞University of Pennsylvania

Background – Steered Query Optimizers
► Database Systems expose knobs that can be used to steer query execution. For example, PostgreSQL has knobs to disable nested loop joins or index scans.
► Hint-sets (HS) can combine multiple knobs.
► Recent work on steered query optimizers either predefined [1] or randomly chooses [2, 3] multiple hint-sets, which are used to generate alternative query plans.
► A deep neural network predicts the execution time of each plan.

Greedy Hint-Set Exploration
► 2^#knobs different hint-sets; However, most yield bad plans.
► Greedy search finds promising hint-sets with reasonable overhead.
► Assumption:
  Larger, beneficial HSs consist of smaller, beneficial HSs.
  Not always, but in many cases true. Experimentally tested.
► Input: SQL query and query span (example: [k1, k3, k5, k9])

Inference Mode ( PrestoDB / Join Order Benchmark )
► AutoSteer with an integrated database connector for PrestoDB.
► AutoSteer using a tree convolutional neural network to infer execution times at runtime.
► Reduces execution times of unseen queries by 20.6% (opaque) and seen queries by 26.8% (transparent).

Training Mode ( PrestoDB / Join Order Benchmark )
► Focus on tail latencies.
► > 3000 queries scanning petabytes of data.
► Workload runs every day on a PrestoDB cluster having hundreds of compute nodes.

Dashboard Application at Meta ( PrestoDB )
Best Known Plan
AutoSteer’s Inference Mode
PrestoDB

Query & Hint-Set Evaluation
Query Span
Greedy HS Search
TCNN
Notation
Training mode
Inference mode
All modes
Database-specific

SQL Queries
DBMS
Parser
Logical Optimizer
Execution Engine

System Overview
AutoSteer-G
Apply a hint-set
DBMS
Config
Ir
Logical Optimizer
exec

Green Hint-Set Exploration
► 2^#knobs different hint-sets; However, most yield bad plans.
► Greedy search finds promising hint-sets with reasonable overhead.
► Assumption:
  Larger, beneficial HSs consist of smaller, beneficial HSs.
  Not always, but in many cases true. Experimentally tested.
► Input: SQL query and query span (example: [k1, k3, k5, k9])