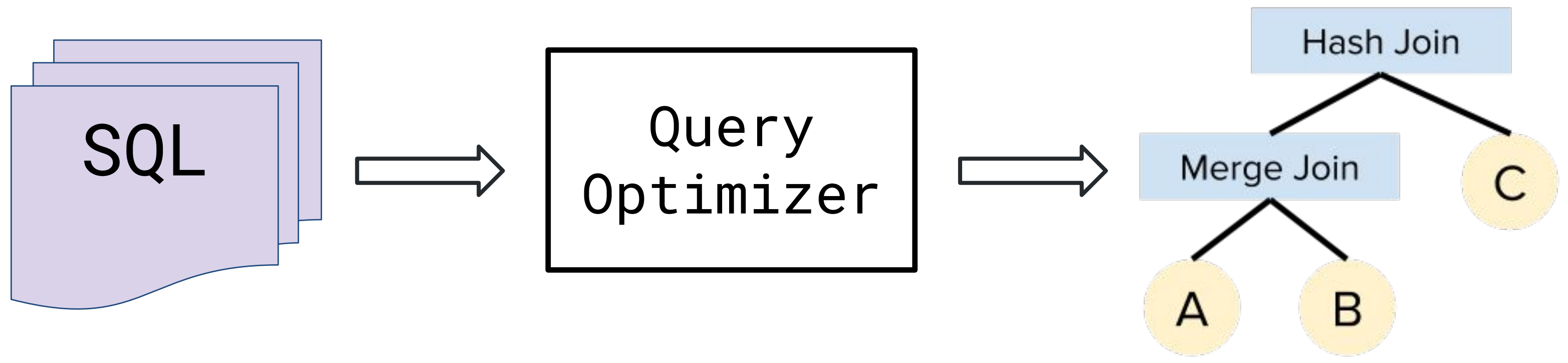
Balsa: Learning a Query Optimizer Without Expert Demonstrations

riselab UC Berkeley

Zongheng Yang, Wei-Lin Chiang*, Frank Luan*,
Gautam Mittal, Michael Luo, Ion Stoica

Optimizers take expert-years to develop & maintain

Optimizers are responsible for producing the best execution plan for a declarative query:



As a performance-critical component, they have been **costly to develop or maintain:**

First optimizer since pre-2000s
Commits to optimizer till occurring

Cockroach DB — Shipped first optimizer by a team and *"9 months of intense effort"*

Spark SQL — Heuristic optimizer in 2014
Cost-based opt. 3 years later

New data systems lack an "expert" to imitate from

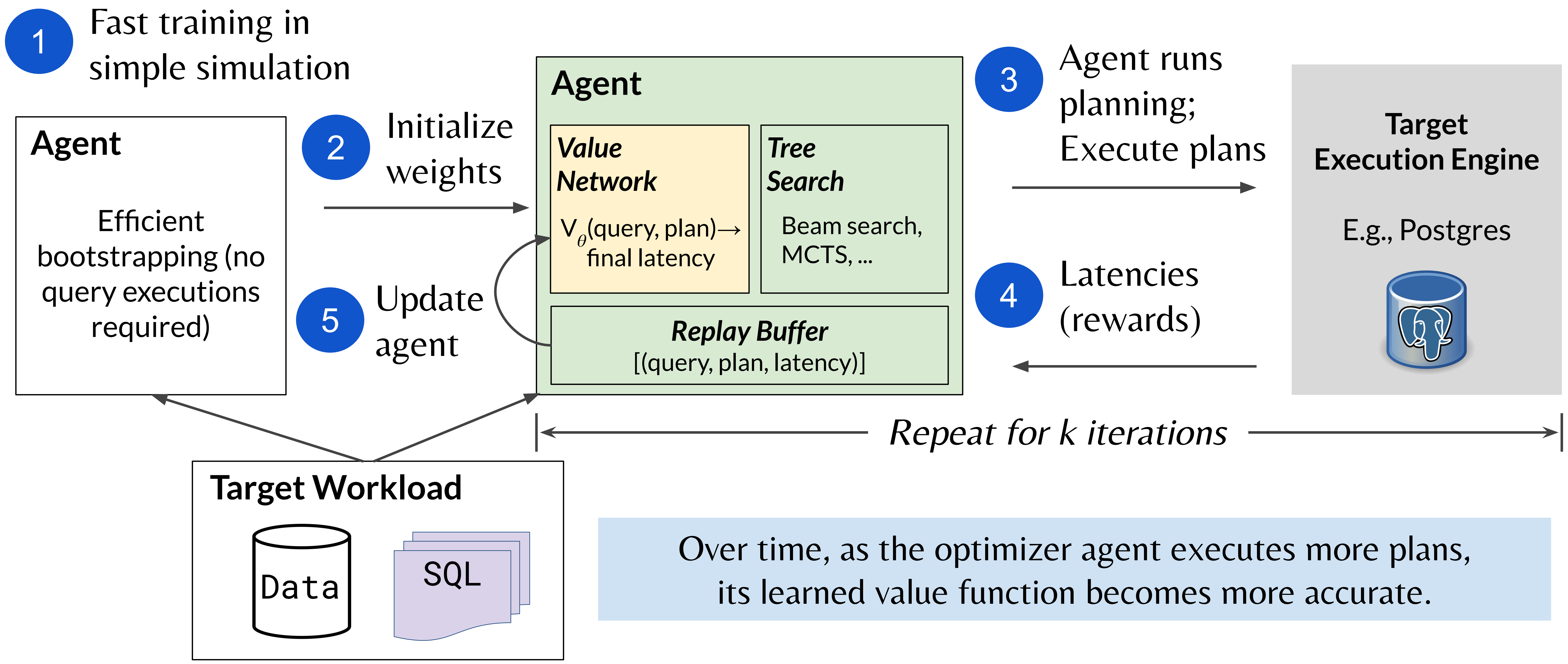
RAY — New execution engine/model (dynamic tasks + actors)

Materialize — New objective: lower memory footprint vs. latency

rAI MODIN NumS

Balsa: a Learned Query Optimizer

Key idea: Without imitating an existing expert optimizer, learn by trial-and-error using deep RL

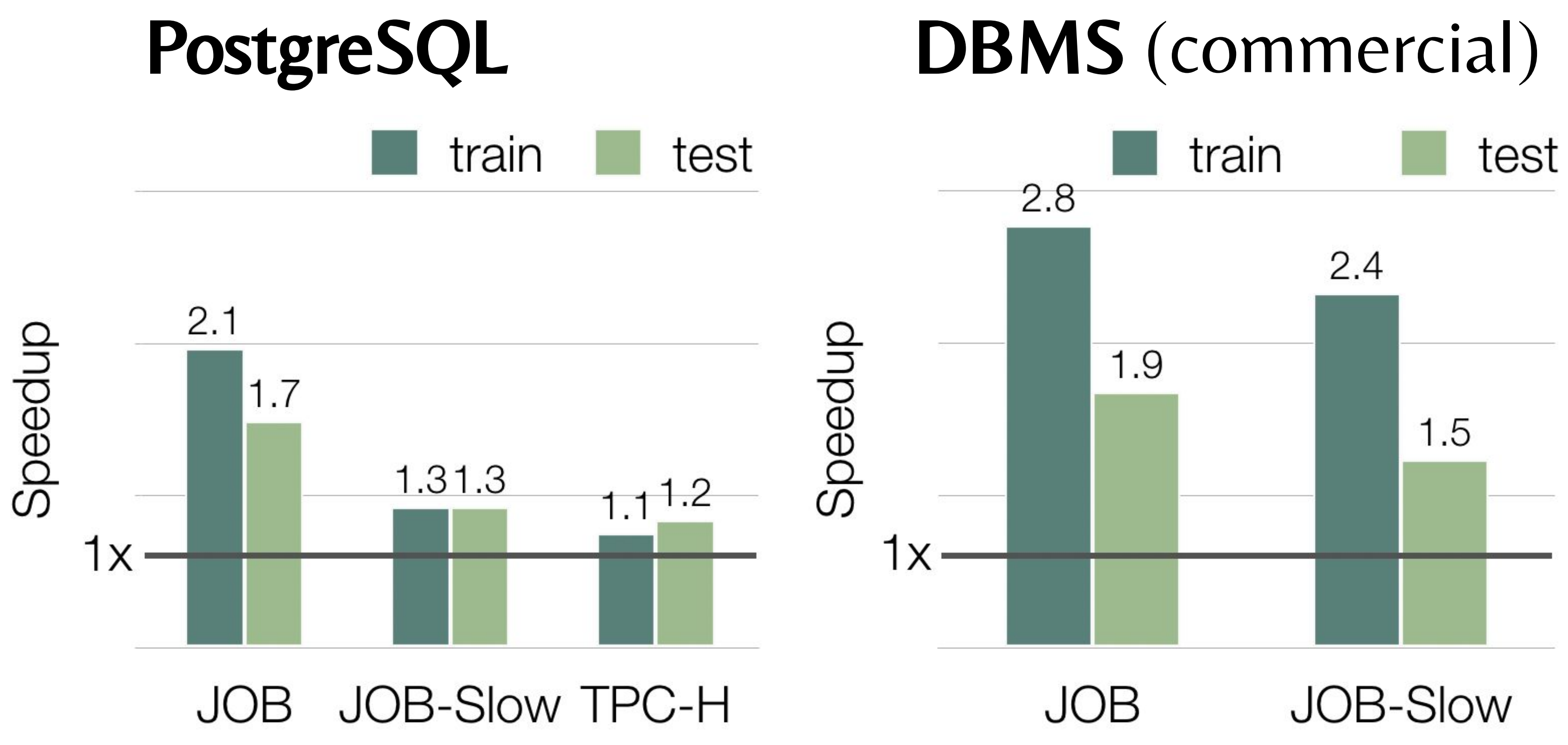


Over time, as the optimizer agent executes more plans, its learned value function becomes more accurate.

Key challenges

Cold start	Without imitating an expert, a cold-start agent can take forever to learn
Bad actions are slow	Many bad plans, which take too long to run (in games, bad actions speed up episodes)
Exploration	Ensuring the exponential search space is explored sufficiently

Evaluation



Takeaway: Balsa outperforms two mature expert systems & generalizes to unseen queries

We'd love to chat: {zongheng,lsf,weichiang}@cs.berkeley.edu