Optimizing Join Queries using Deep Reinforcement Learning

EAD-DBMS project

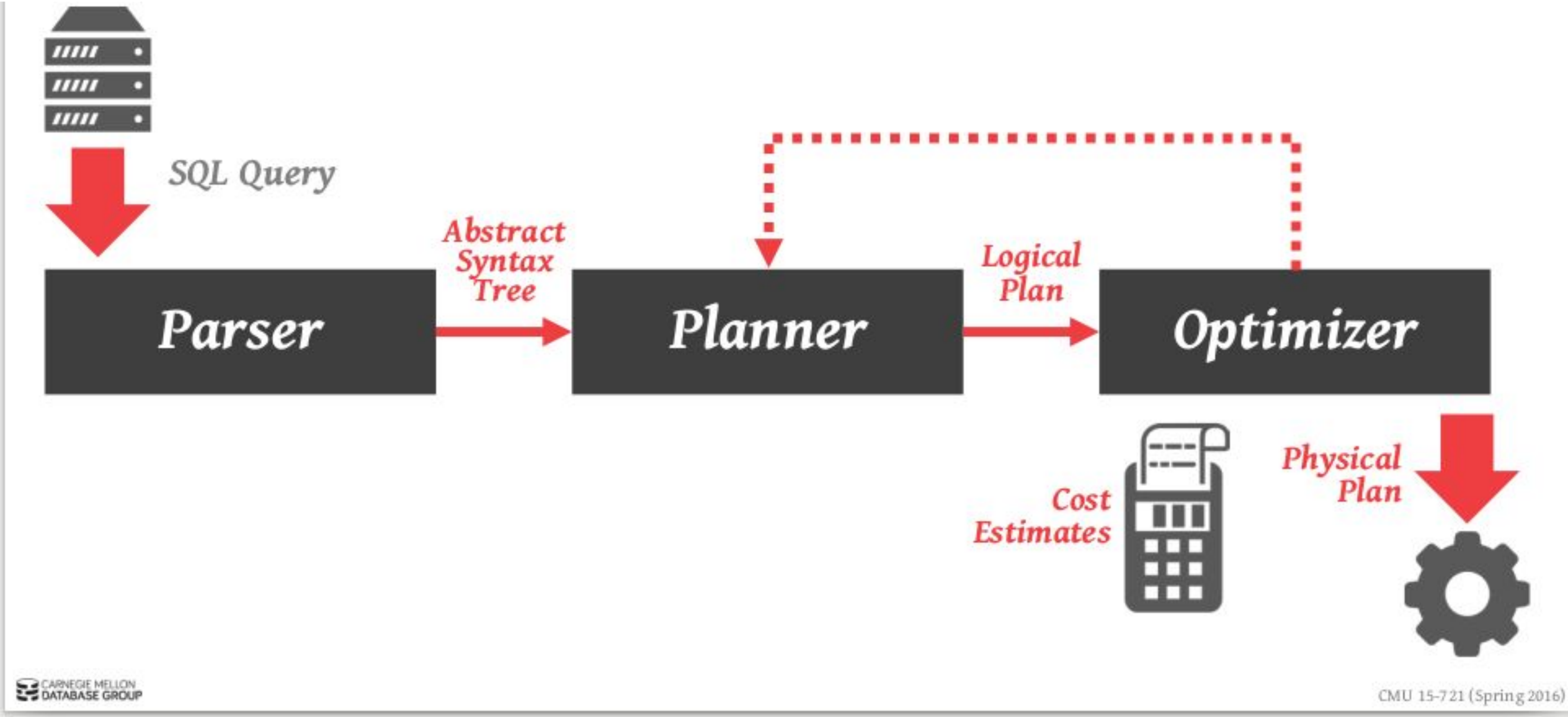
- Amrit Preet Singh 17103090
- Pranshu Mahajan 17103094
- Gauravdeep Singh 17103091

Databases everywhere

The Query Optimization Problem

- Time- Quality trade off
- Parameters of optimization:
  - number of I/O operations required,
  - Execution time,
  - amount of disk buffer space,
  - disk storage service time,

Classic Query optimizers architecture



Motivation for the Project

How Good Are Query Optimizers, Really?

Viktor Leis
TUM
leis@in.tum.de

Andrey Gubichev
TUM
gubichev@in.tum.de

Atanas Mirchev
TUM
mirchev@in.tum.de

Peter Boncz
CWI
p.boncz@cwi.nl

Alfons Kemper
TUM
kemper@in.tum.de

Thomas Neumann
TUM
neumann@in.tum.de

ABSTRACT

Finding a good join order is crucial for query performance. In this paper, we introduce the Join Order Benchmark (JOB) and experimentally revisit the main components in the classic query optimizer architecture using a complex, real-world data set and realistic multi-join queries. We investigate the quality of industrial-strength cardinality estimators and find that all estimators routinely produce large errors. We further show that while estimates are essential for finding a good join order, query performance is unsatisfactory if the query engine relies too heavily on these estimates. Using another set of experiments that measure the impact of the cost model, we find that it has much less influence on query performance than the cardinality estimates. Finally, we investigate plan enumeration techniques comparing exhaustive dynamic programming with heuristic algorithms and find that exhaustive enumeration improves performance despite the sub-optimal cardinality estimates.

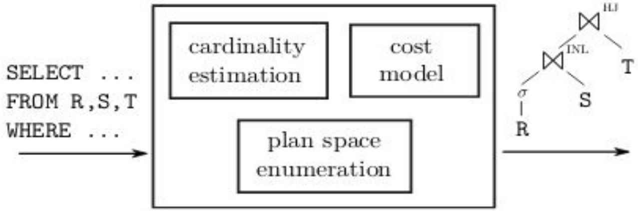
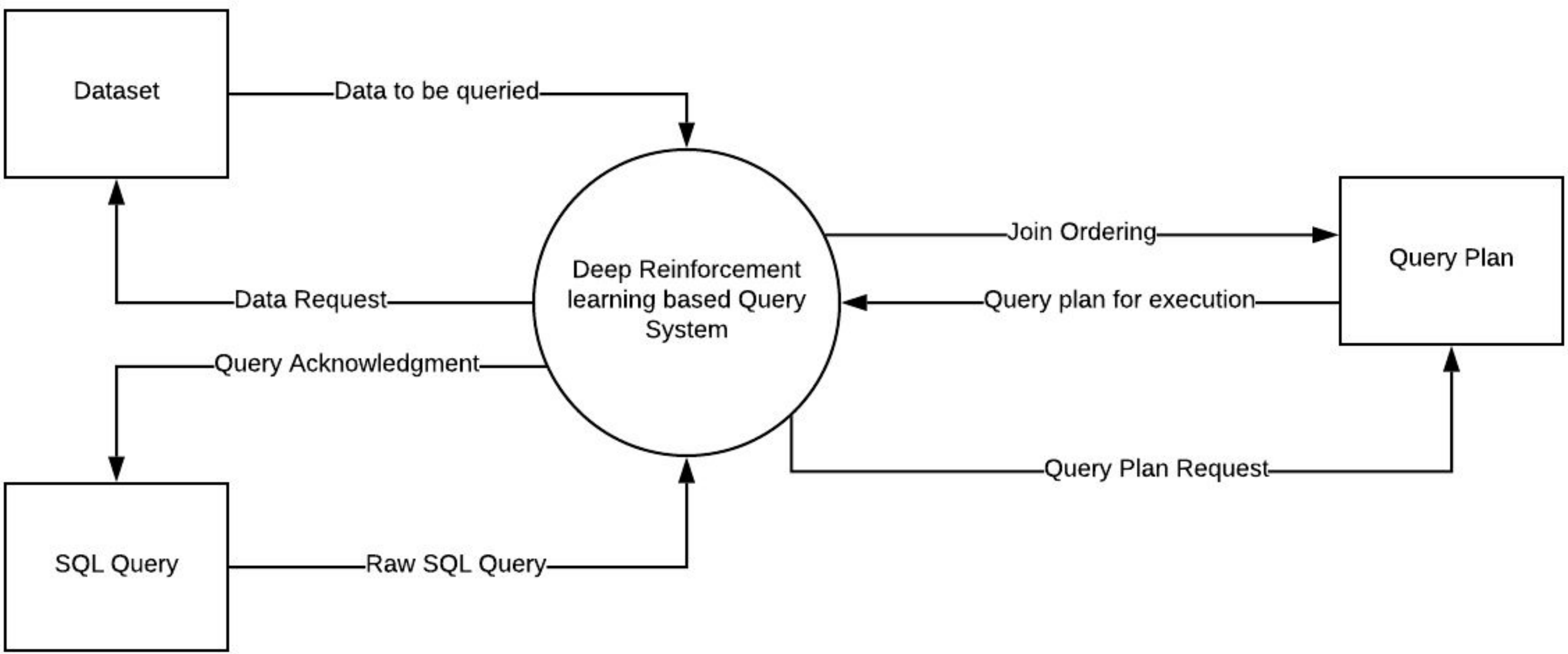
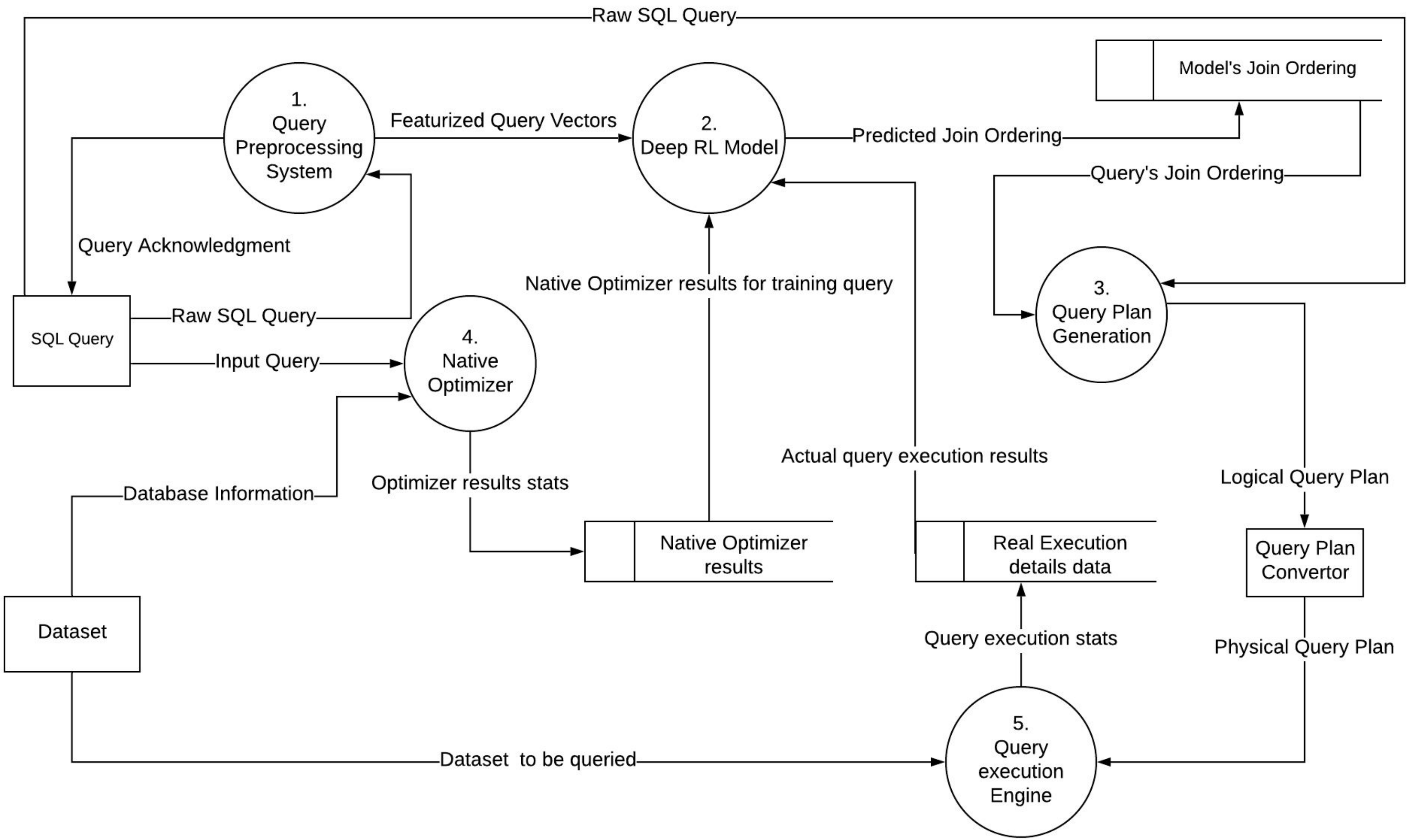


Figure 1: Traditional query optimizer architecture

- How important is an accurate cost model for the overall query optimization process?
- How large does the enumerated plan space need to be?

Proposed Solution




Raw SQL Query
Model's Join Ordering
1.
Query
Preprocessing
System
Featurized Query Vectors
2.
Deep RL Model
Predicted Join Ordering
Query's Join Ordering
Query Acknowledgment
Native Optimizer results for training query
3.
Query Plan
Generation
SQL Query
Raw SQL Query
Input Query
4.
Native
Optimizer
Actual query execution results
Logical Query Plan
Database Information
Optimizer results stats
Native Optimizer
results
Real Execution
details data
Query Plan
Convertor
Dataset
Query execution stats
Physical Query Plan
5.
Query
execution
Engine
Dataset to be queried

Data Dictionaries

1. Data Structure Dictionary
2. Data flow Dictionary
3. Data Store Dictionary
4. Data Process Dictionary

Data Structure Dictionary

S. No.	DATA STRUCTURE	DESCRIPTION	CONTENT	VOLUME
1	Raw SQL Query = Input Query	SQL query in its syntactical form.	Query number | Query offset | Query syntax text	200 for training
2	Featurized Query Vectors	One hot encoding of the query. Concatenation of vectors representing Query Graph, Left side of join, right side of join.	Query number | One hot vectors.	Around 200
3	Database Information	Statistics of the IMDB databse being used.MySQL Workbench gets it automatically.	Database schemas | Tables info | correaltions data |	40
4	Optimizer results stats = Native Optimizer results for training query	Predicted Join Ordering of the native optimizer.	Query number | Join ordering info	200
5	Predicted Join Ordering = Query's Join Ordering	RL model's predicted join ordering	Query Number | **Q value** | |oin Ordering info	200

S. No.	DATA STRUCTURE	DESCRIPTION	CONTENT	VOLUME
6	Generated Query Plan	Logical query Plan information for given query.	Query Number | Apache Calcite query planner object	~200
7	Query plan details	Physical Query Plan Information for given query	Query Number | Executable Query Plan Object	~200
8	Query execution stats	Stats of the actual exxecution of the query	Query Number | output of EXPLAIN statement	~200
9	Dataset to be queried	IIMDb database	Whole IMDb database	~1
10	Query Acknowledgment	Return value of Check for correct syntax of SQL query	[True | false]	

Data Flow Dictionary

S. No.	DATA FLOW NAME	DESCRIPTION	From	To	Data Structures
1.	Raw SQL Query	SQL query in its syntactical form.	SQL Query	1.Query Preprocessing System 2. Native optimizer	Raw SQL Query
2	Featurized Query Vectors	One hot encoding of the query.	Query Preprocessing System	Deep RL Model	Featurized Query Vectors
3	Database Information	Statistics of the IMDB databse	Dataset	Native Optimizer	Database Information
4	Predicted Join Ordering (= Query's Join Ordering)	RL model's predicted join ordering	1. Deep RL Model 2. Model's Join Ordering	1. Model's Join Ordering 2. Query Plan Generation	Predicted Join Ordering
5.	Generated Query Plan	Logical query Plan information	Query Plan Generation	Query Plan	Generated Query Plan

S. No.	DATA FLOW NAME	DESCRIPTION	From	To	Data Structures
6	Optimizer results stats (= Native Optimizer results for training query)	Predicted Join Ordering of the native optimizer.	1. Native Optimizer 2. Native Optimize rresults	1. Native Optimizer results 2. Deep RL Model	Optimizer results stats Native, Optimizer results for training query
7.	Query plan details	Physical Query Plan Information	Query Plan	Query execution Engine	Query plan details
8.	Query execution stats	Stats of the actual exxecution of the query	Query execution Engine	Real Execution details data	Query execution stats
9.	Dataset to be queried	IMDb database	Dataset	Query execution Engine	Dataset to be queried
10	Query Acknowledgment	Return value of Check for correct syntax of SQL query	Query Preprocessing System	SQL Query	Query Acknowledgment

Data Store Dictionary

S. No	DATA STORE	DESCRIPTION	Inbound	Outbound	Data description	Volume
1	Native Optimizer results	Stores output of native optimizer	Optimizer results stats	Native Optimizer results for training query	Query Number | Join Ordering	200 tuples
2.	Model's Join Ordering	Output of the Deep RL Model	Predicted Join Ordering	Query's Join Ordering	Query Number | **Q value** | |oin Ordering info	200 tuples
3.	Real Execution details data	Stats of the actual exxecution of the query are stored.	Query execution stats	Actual query execution results	Query Number | output of EXPLAIN statement	200 tuples

Process Dictionary

S. No	PROCESS	DESCRIPTION	Input	Output	Logic summary
1.	Query Preprocessing System	Preprocesses the SQL query to be in featurized vector form	Raw SQL Query	> Featurized Query Vectors > Query Acknowledgment	Use One Hot encoding for Query Graph, Join participating relations
2.	Deep RL Model	Q-Value function approximator	> Featurized Query Vectors > Native Optimizer results for training query > Actual query execution results	> Predicted Join Ordering	Use Deep multilayered Neural network to approximate the Q-Value Funtion
3.	Query Plan Generation	Generates logical query plan from predicted join order.	> Query's Join Ordering > Raw SQL query	Generated Query Plan	Use the Apache Calcite tool tto generate logical plan.

S. No	PROCESS	DESCRIPTION	Input	Output	Logic summary
4.	Query execution Engine	Executes the query plan	> Query Plan details > Dataset to be queried	Query execution stats	Mysql integrated with Apache Calcite
5.	Native Optimizer	**MySQL query** optimizer.	> Input Query > Database Information	Optimizer results stats	Inbuilt MySQL Optimizers.

Mathematical Proof of Optimality

Workflow:

1. Data Collection. ✓
2. Training data generation. ✓
3. Deciding model architecture and choosing the tools. ✓
4. Training the model.
5. Testing with Join Order Benchmark.
6. Post testing improvements.
7. Integrating into currently used optimizers.(optional).

Thank You