Special Topics:
Self-Driving Database Management Systems

Behavior Modeling II

@William Zhang // 15-799 // Spring 2022
Model via independent operating units

**OU Examples**
- Build Hash Table
- Create Index
- Serialize Log Records

**Input Features**
- Knobs

**Output Labels**
- [Diagram of output labels]
LAST CLASS (MB2)

- Specialized Runners
- Lightweight Metrics System
- Training Framework
- Behavior Models
  - Operating Unit Models
  - Interference Model
BACKGROUND

DNN Models
CHALLENGES

Query Plan Structure

Diverse Operator Features

Position Independent Operator Behavior
QPPNet

Plan Structured Neural Network
→ Operator Neural Units
→ Tree of Operator Neural Units
OPERATOR NEURAL UNIT

- Captures a distinct operator
- Outputs latency prediction and opaque vector
OPERATOR NEURAL UNIT (NON-LEAF)

- Incorporates outputs of child operators
### OPERATOR INPUTS

<table>
<thead>
<tr>
<th>Feature</th>
<th>PostgreSQL operators</th>
<th>Encoding</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plan Width</td>
<td>All</td>
<td>Numeric</td>
<td>Optimizer’s estimate of the width of each output row</td>
</tr>
<tr>
<td>Plan Rows</td>
<td>All</td>
<td>Numeric</td>
<td>Optimizer’s estimate of the cardinality of the output of the operator</td>
</tr>
<tr>
<td>Plan Buffers</td>
<td>All</td>
<td>Numeric</td>
<td>Optimizer’s estimate of the memory requirements of an operator</td>
</tr>
<tr>
<td>Estimated I/Os</td>
<td>All</td>
<td>Numeric</td>
<td>Optimizer’s estimate of the number of I/Os performed</td>
</tr>
<tr>
<td>Total Cost</td>
<td>All</td>
<td>Numeric</td>
<td>Optimizer’s cost estimate for this operator, plus the subtree</td>
</tr>
<tr>
<td>Join Type</td>
<td>Joins</td>
<td>One-hot</td>
<td>One of: semi, inner, anti, full</td>
</tr>
<tr>
<td>Parent Relationship</td>
<td>Joins</td>
<td>One-hot</td>
<td>When the child of a join. One of: inner, outer, subquery</td>
</tr>
<tr>
<td>Hash Buckets</td>
<td>Hash</td>
<td>Numeric</td>
<td># hash buckets for hashing</td>
</tr>
<tr>
<td>Hash Algorithm</td>
<td>Hash</td>
<td>One-hot</td>
<td>Hashing algorithm used</td>
</tr>
<tr>
<td>Sort Key</td>
<td>Sort</td>
<td>One-hot</td>
<td>Key for sort operator</td>
</tr>
<tr>
<td>Sort Method</td>
<td>Sort</td>
<td>One-hot</td>
<td>Sorting algorithm, e.g., “quicksort”, “top-N heapsort”, “external sort”</td>
</tr>
<tr>
<td>Relation Name</td>
<td>All Scans</td>
<td>One-hot</td>
<td>Base relation of the leaf</td>
</tr>
<tr>
<td>Attribute Mins</td>
<td>All Scans</td>
<td>Numeric</td>
<td>Vector of minimum values for relevant attributes</td>
</tr>
<tr>
<td>Attribute Medians</td>
<td>All Scans</td>
<td>Numeric</td>
<td>Vector of median values for relevant attributes</td>
</tr>
<tr>
<td>Attribute Maxxs</td>
<td>All Scans</td>
<td>Numeric</td>
<td>Vector of maximum values for relevant attributes</td>
</tr>
<tr>
<td>Index Name</td>
<td>Index Scans</td>
<td>One-hot</td>
<td>Name of index</td>
</tr>
<tr>
<td>Scan Direction</td>
<td>Index Scans</td>
<td>Boolean</td>
<td>Direction to read the index (forward or backwards)</td>
</tr>
<tr>
<td>Strategy</td>
<td>Aggregates</td>
<td>One-hot</td>
<td>One of: plain, sorted, hashed</td>
</tr>
<tr>
<td>Partial Mode</td>
<td>Aggregate</td>
<td>Boolean</td>
<td>Eligible to participate in parallel aggregation</td>
</tr>
<tr>
<td>Operator</td>
<td>Aggregate</td>
<td>One-hot</td>
<td>The aggregation to perform, e.g. max, min, avg</td>
</tr>
</tbody>
</table>

Table 2: QPP Net Inputs
Insights

→ Non-independence (child → all ancestors)
→ Handle diverse operator input features
→ Single “Unit” per operator
TRAINING PIPELINE

Workloads → Executed Plans → Learning → Scan → IndexScan
Minimize Latency Error

- Across all operators simultaneously
- Neural Unit Data Vector is "free-floating"
- Enables modeling dependence
EVALUATION

Workloads
→ TPC-H, TPC-DS (SF of 100GB)
→ 22 TPC-H templates, 69 TPC-DS templates
→ 22K queries

Train/Test Split
→ TPC-H 90%/10% data split
→ TPC-DS withhold 10 query templates
→ With all optimizations, takes ~24 hours
EVALUATION (PREDICTION)

Baselines
→ SVM w. hand-picked features
→ RBF w. hand-picked features
→ TAM (“tuned cost model”)
→ DNN (non-tree deep neural network)
EVALUATION (ERROR DISTRIBUTION)

Figure 7: Prediction error distributions for TPC-DS

- Normalized to sum to 1
- QPPNet narrow, center error bounds
EVALUATION (CACHE)

→ Execution data from cold/warm cache
→ Cache state is not an input to model
→ QPPNet Error: Cold (17%), Warm (19%)
EVALUATION (PARALLEL)

Special MP Neural Unit

Concurrent Query #1
Concurrent Query #2
Concurrent Query #3
EVALUATION (PARALLEL)

→ TPC-DS (2346 pairs, 52394 triples)
→ Comparison: BAL (SOTA predict buffer access latency)

(c) Prediction error per concurrency levels
→ Train at 1 DB size and test at other sizes (shrink v. expand)
→ Resistant to skew but skew transfer not tested
TAKEAWAYS

Plan Structured Neural Networks

→ Capture plan structure and operator interactions
→ Formulate complex (opaque) relationships between inputs

Comments

→ Workload “embedding” into the models
→ Questionable generalizability (or at least DB size)
→ “Interference Model”-approach to parallel
→ No direct incorporation of cache state
→ Targeting OLAP (TPC-H, TPC-DS)