**Optimal Join Order Selection**

**Goal:** selecting an optimal order of relations involved in a given query so that the system consumes less amount of memory and cpu resources

**Major variables that affects the performance of query execution:**
- accurate selectivities
- cost model
- size of join order search space

**Our Contribution**

**Pre-computation:** represent relations in matrix forms that is build on true selectivities and capture appropriate join attribute values

**Estimation:** estimate join cardinalities via relations’ sketch representations

**Graph-based search:** exaustive depth-first search on each node in join graph

**Fast-AGM Sketch for Join Size Estimation**

- 4-wise independent ±1 random variables
- 2-universal hash function $h: I \rightarrow 1 \cdots n$

**Dataset and Setup**

**IMDB benchmark dataset:** real-world dataset containing correlations and non-uniform data distributions

**Join Order Benchmark:** challenging realistic workload
- multi-join cyclic queries
- 28 join predicates in the most complicated query
- 17 involved relations in the most complicated query

**MapD:** in-memory, GPU-accelerated, column-oriented database system

**Server Environment**
- 2 Intel(R) Xeon(R) CPU E5-2660 v4 @ 2.00GHz
- 1 Tesla K80 GPU
- 8 DDR4 memory 32GB @ 2400 MHZ (total: 256GB)

**Runtime Overhead**

**Intermediate Join Cardinalities**