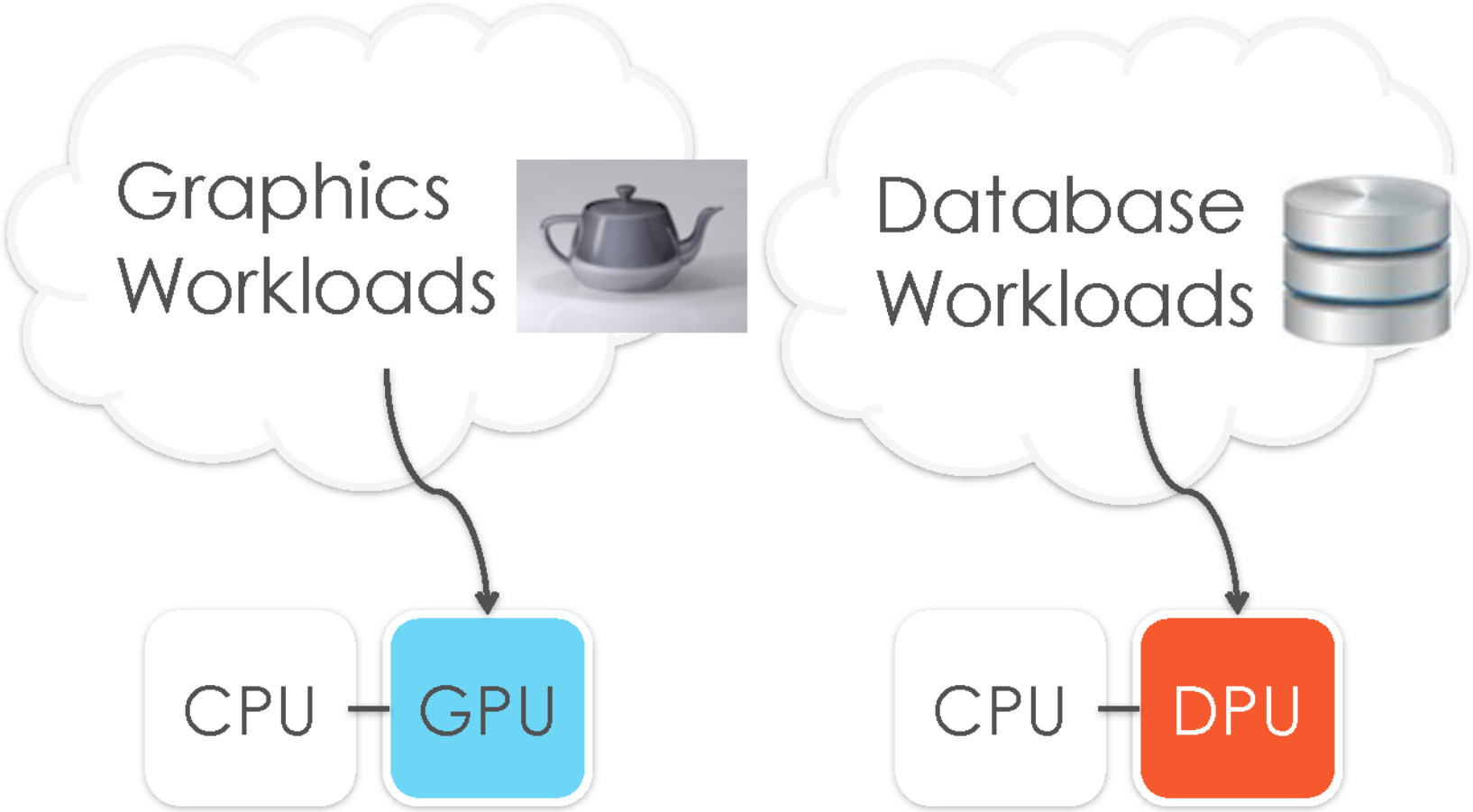
# Q100:
# The Architecture and Design of a DATABASE PROCESSING UNIT

# DPUs are analogous to GPUs



# Q100: A Stream-Based DPU

# Q100: A Stream-Based DPU

- Accelerates analytic queries

# Q100: A Stream-Based DPU

- Accelerates analytic queries

The sale of an airline ticket

Transactional Processing

Sales Projection

Analytic Processing

# Q100: A Stream-Based DPU

- Accelerates analytic queries
- Direct hardware support for relational operators

# Q100: A Stream-Based DPU

- Accelerates analytic queries
- Direct hardware support for relational operators

# Q100: A Stream-Based DPU

- Accelerates analytic queries
- Direct hardware support for relational operators
- Processes data as streams

# Q100: A Stream-Based DPU

- Accelerates analytic queries
- Direct hardware support for relational operators
- Processes data as streams

# Q100: A Stream-Based DPU

- Accelerates analytic queries
- Direct hardware support for relational operators
- Processes data as streams
- Combines spatial and temporal instructions to form a DPU ISA

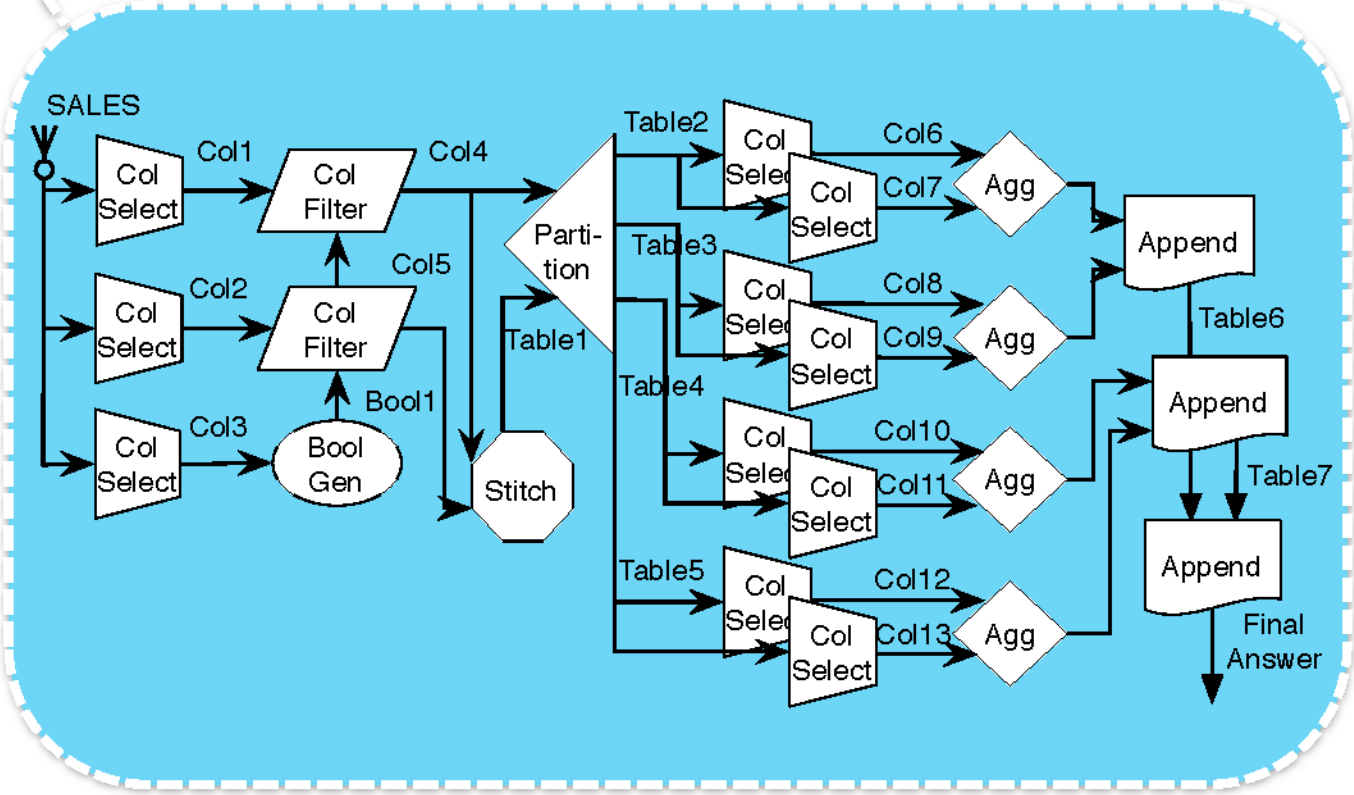
Query

```
SELECT s_season, SUM(s_qty) as sum_qty
FROM sales
WHERE s_shipdate >= '2013-01-01'
GROUP BY s_season
ORDER BY s_season
```

Query

Plan

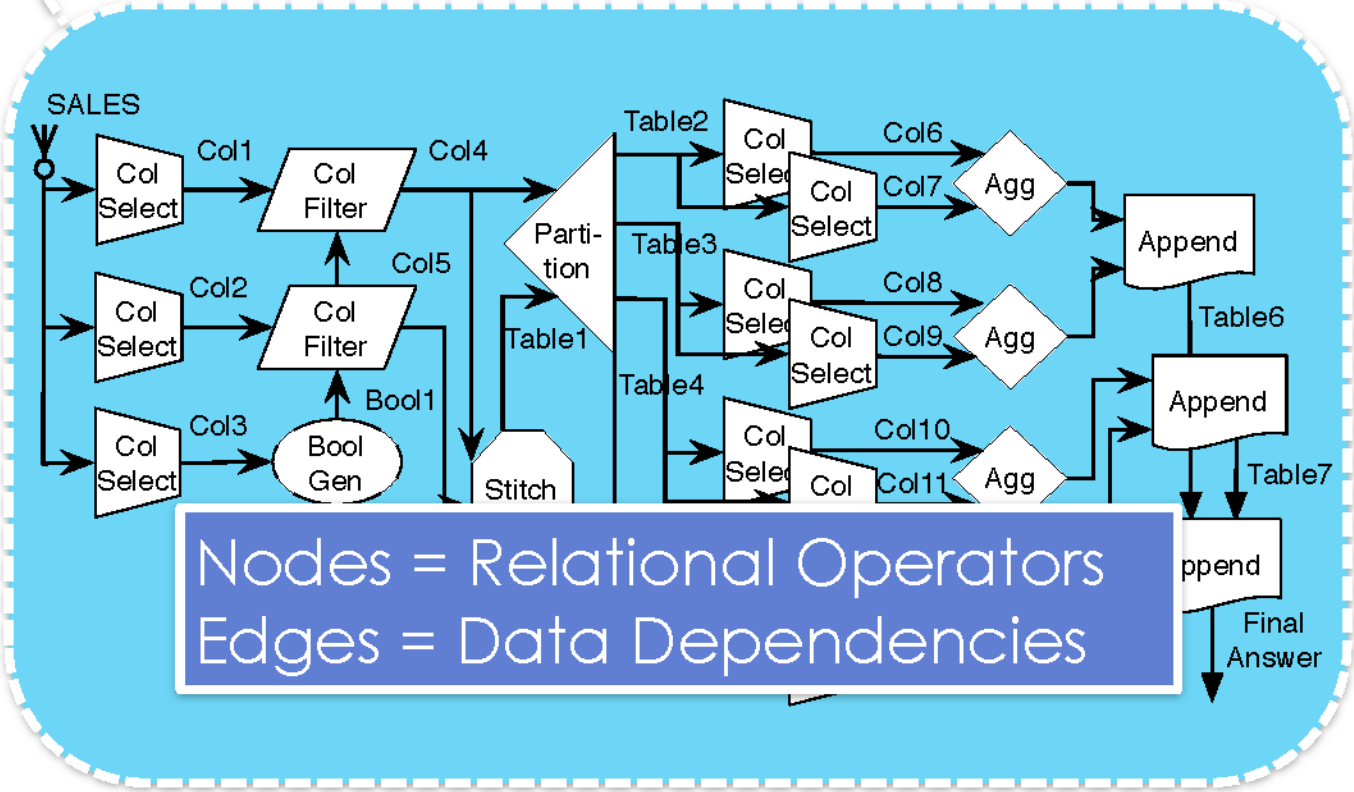
```
SELECT s_season, SUM(s_qty) as sum_qty
FROM sales
WHERE s_shipdate >= '2013-01-01'
GROUP BY s_season
ORDER BY s_season
```



Query

Plan

```
SELECT s_season, SUM(s_qty) as sum_qty
FROM sales
WHERE s_shipdate >= '2013-01-01'
GROUP BY s_season
ORDER BY s_season
```

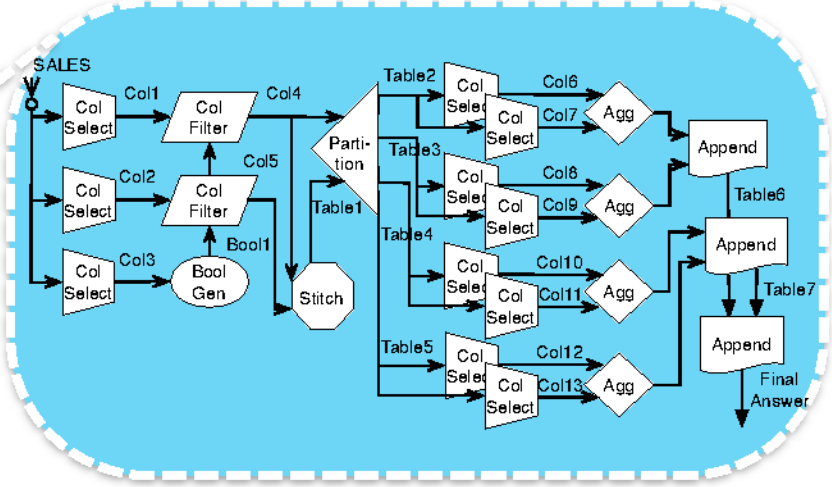


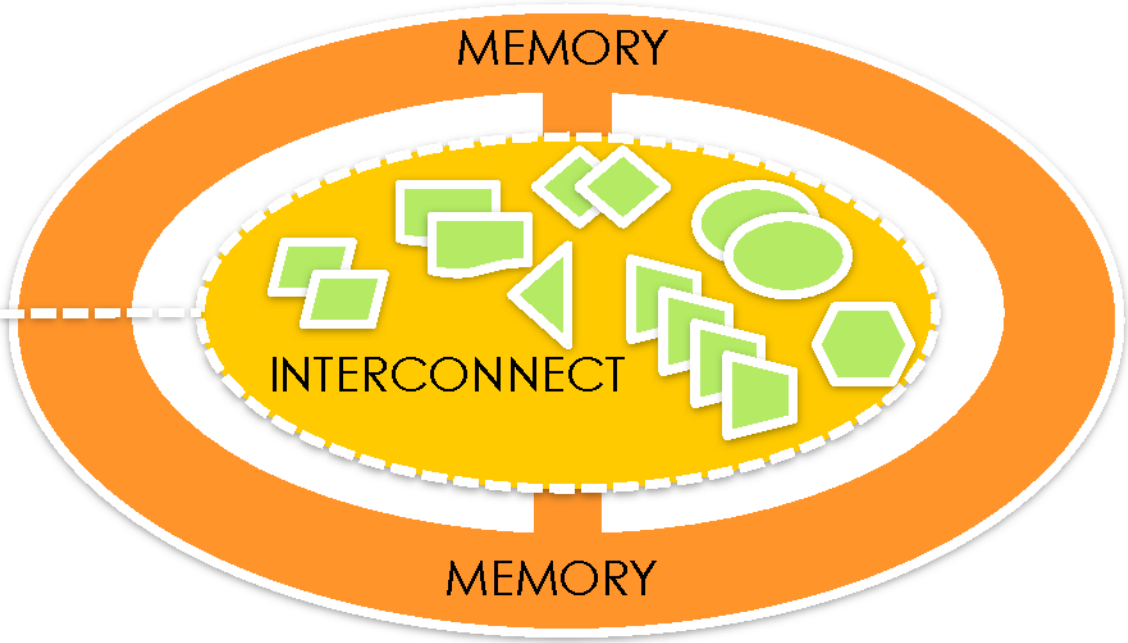
Query

Plan

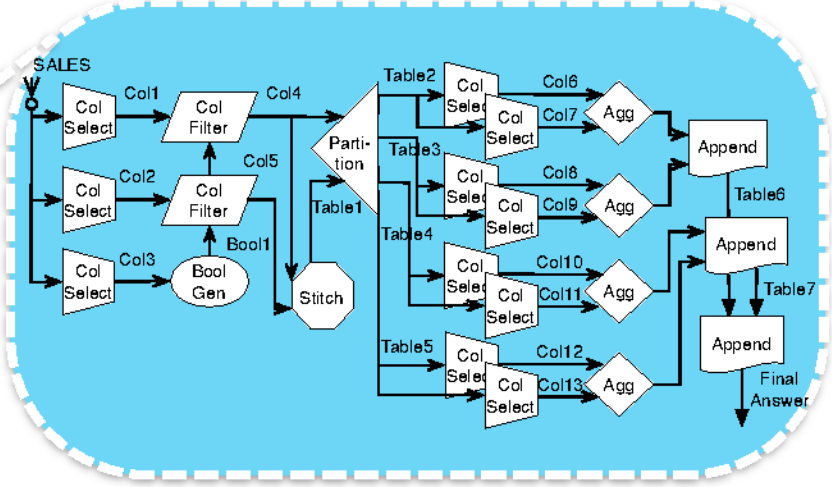
Q100 Program

Q100 Device



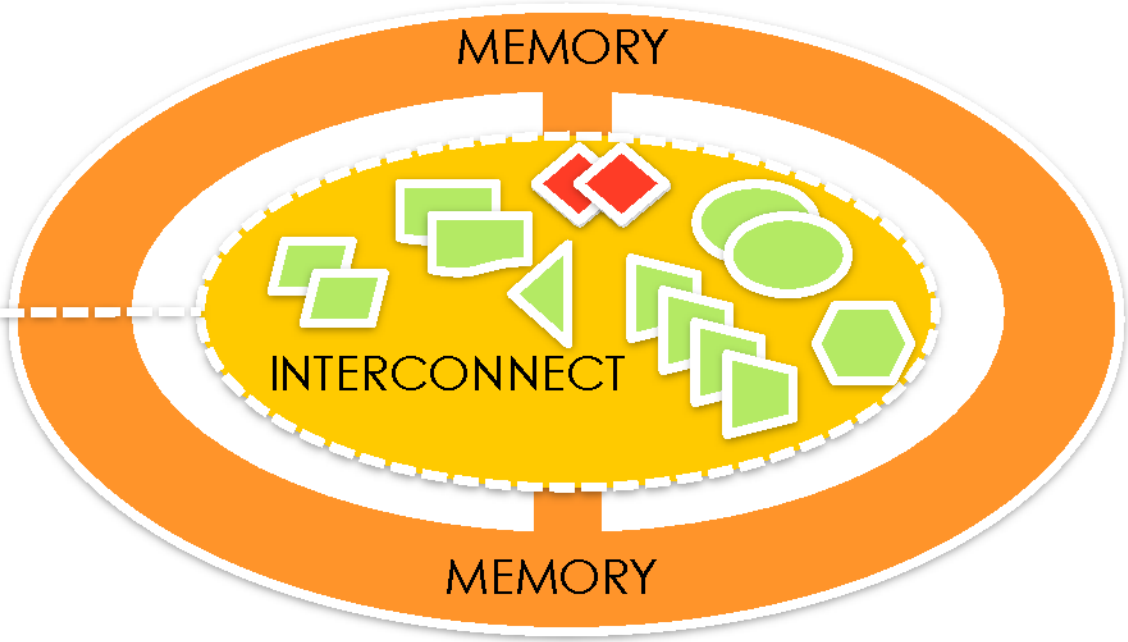
Query



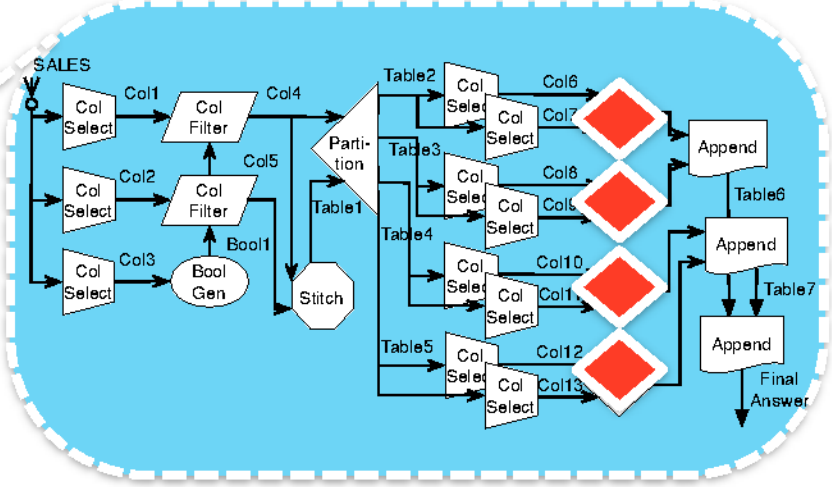

Plan

Q100 Program

Q100 Device



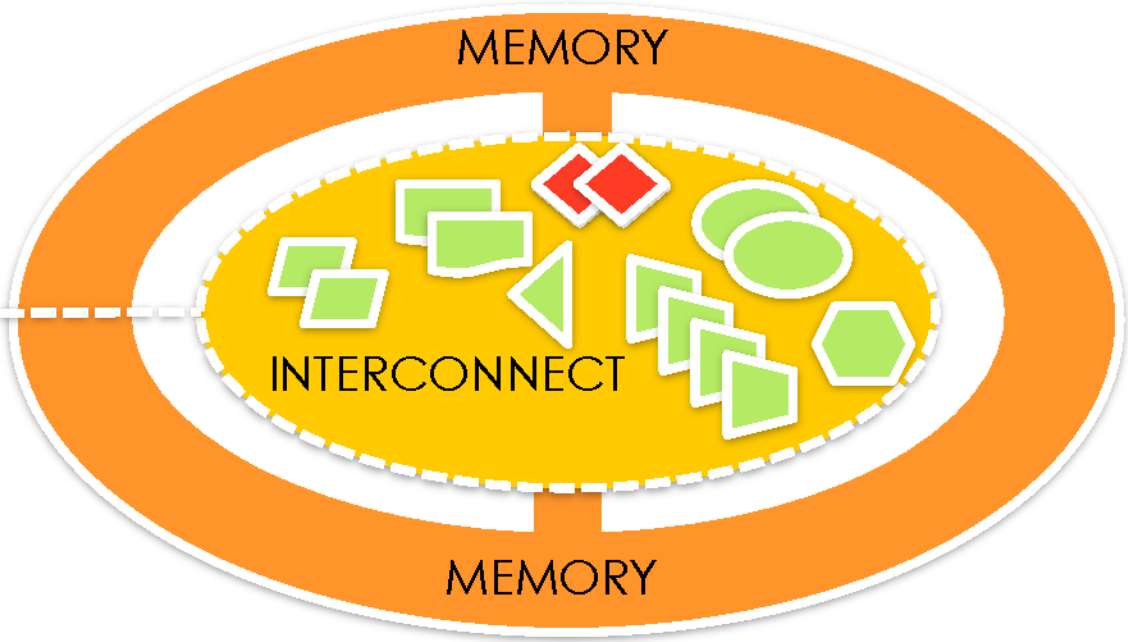
Query



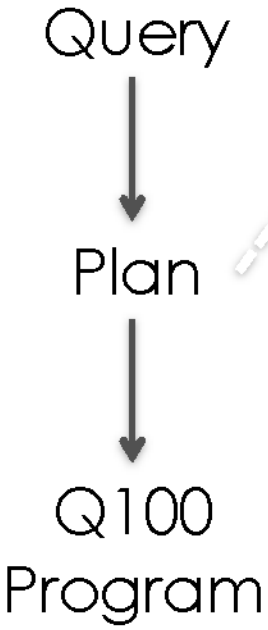

Plan

Q100 Program

Q100 Device



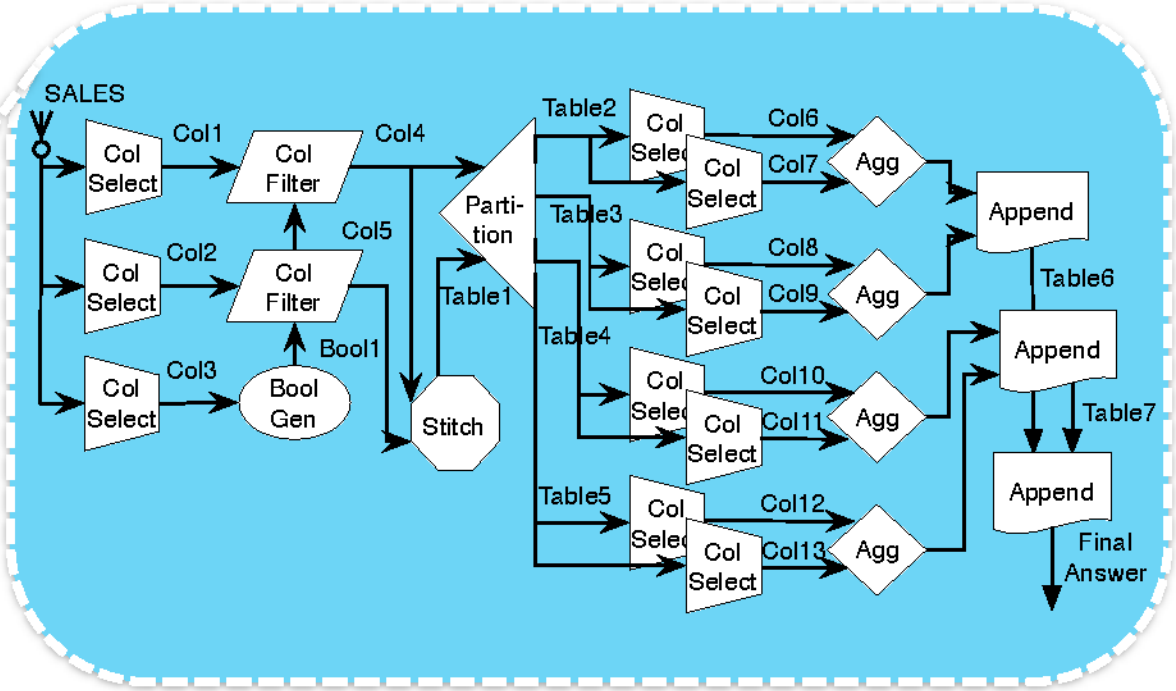
Query

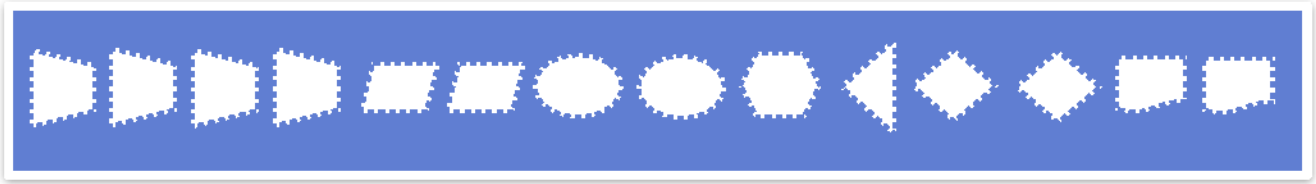


Plan



Q100 Program

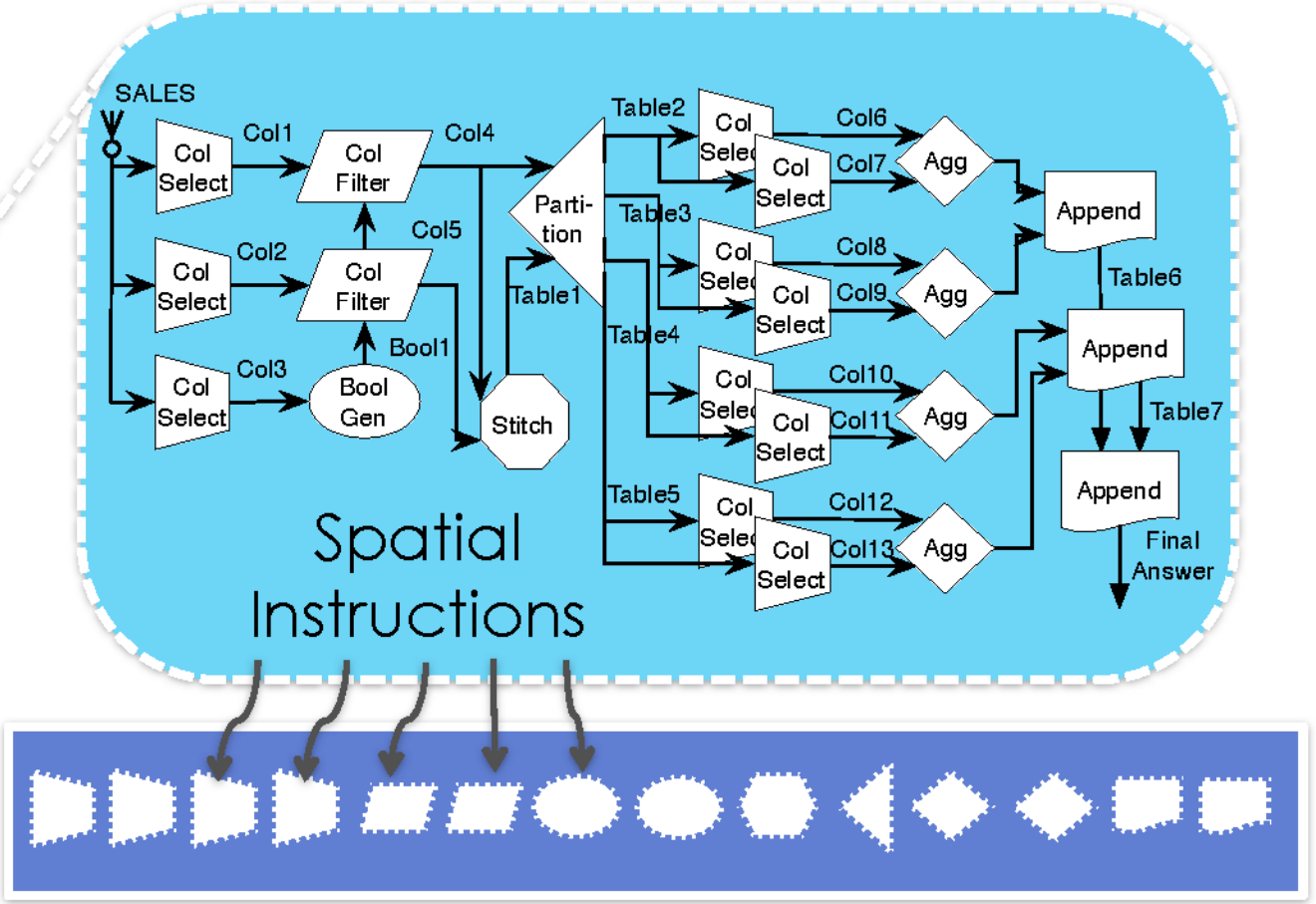


Query

Plan

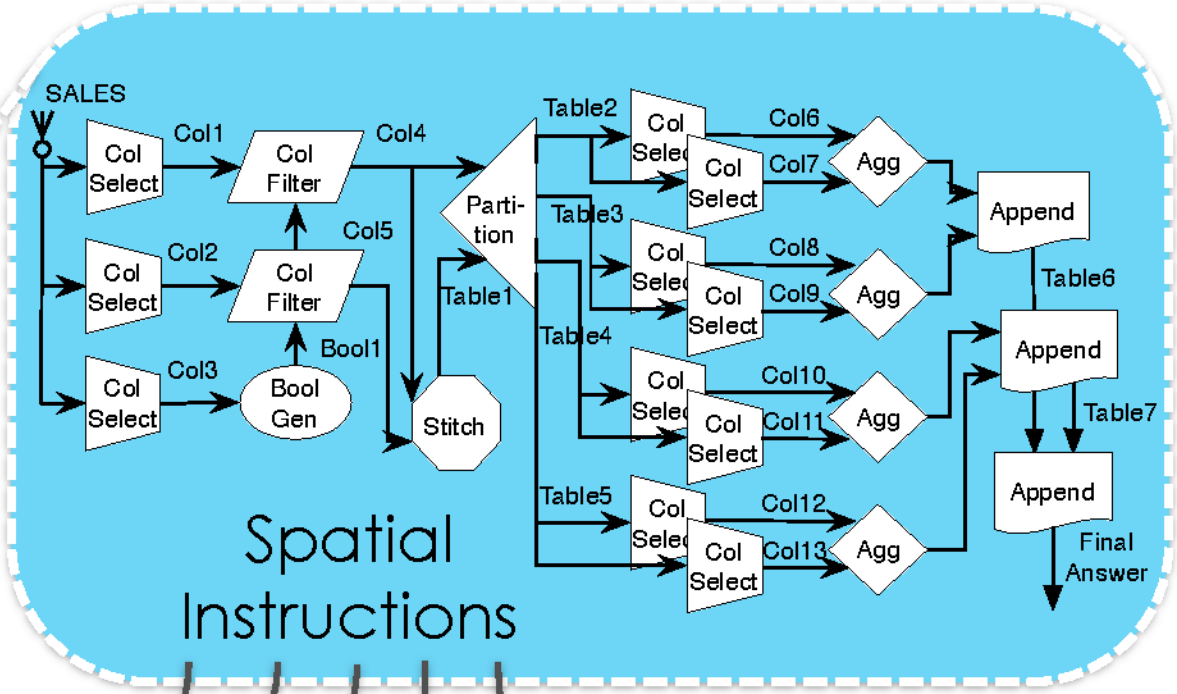
Q100 Program

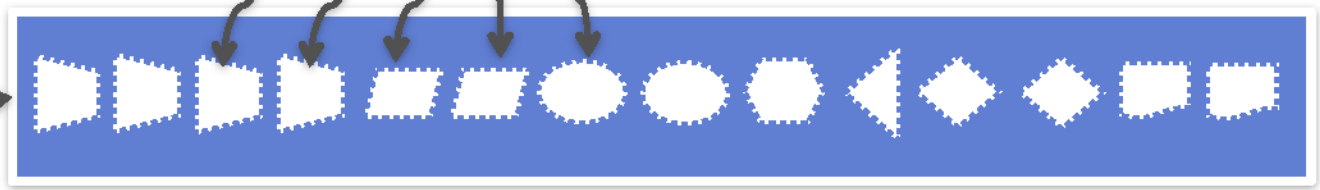


Query

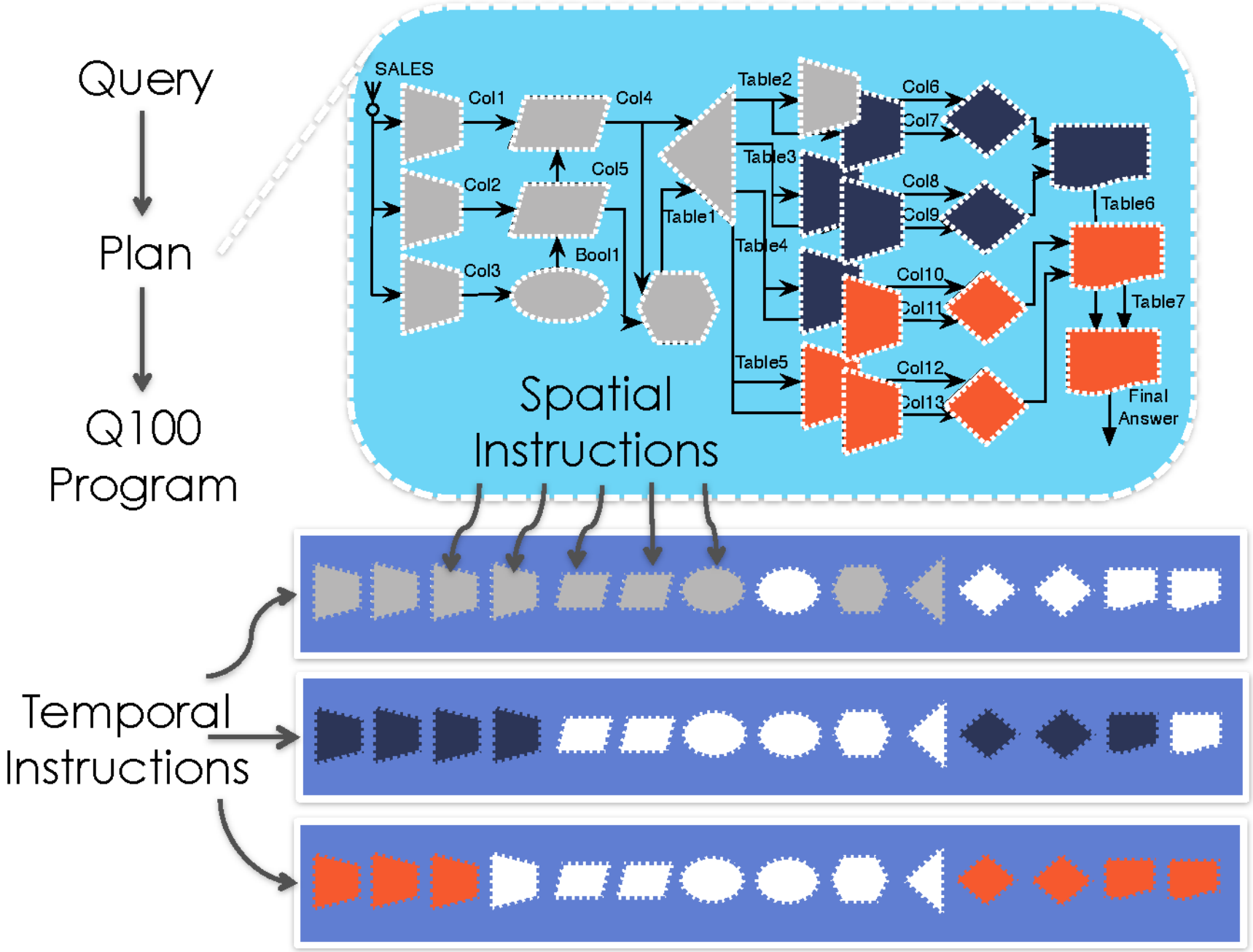
Plan

Q100 Program

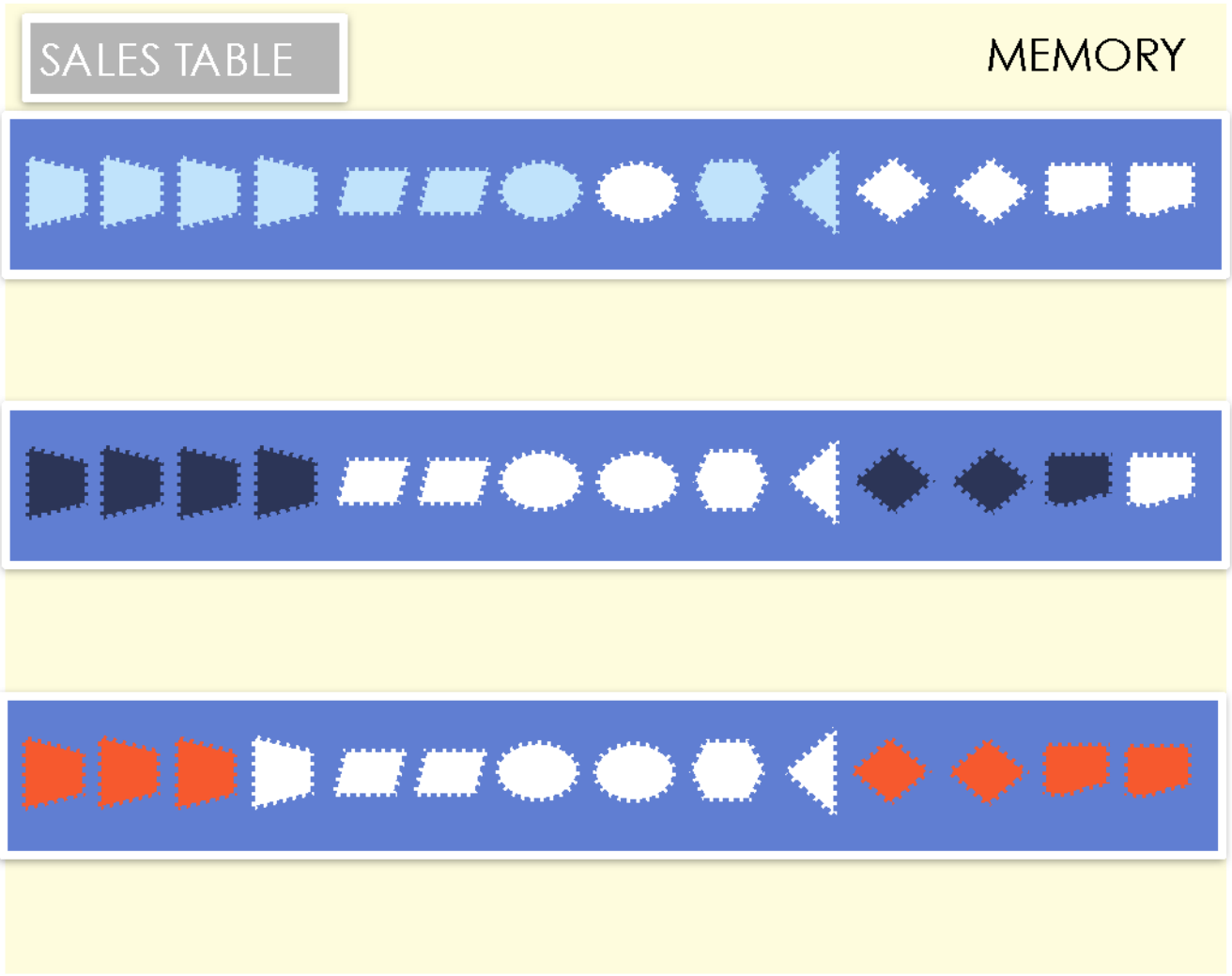


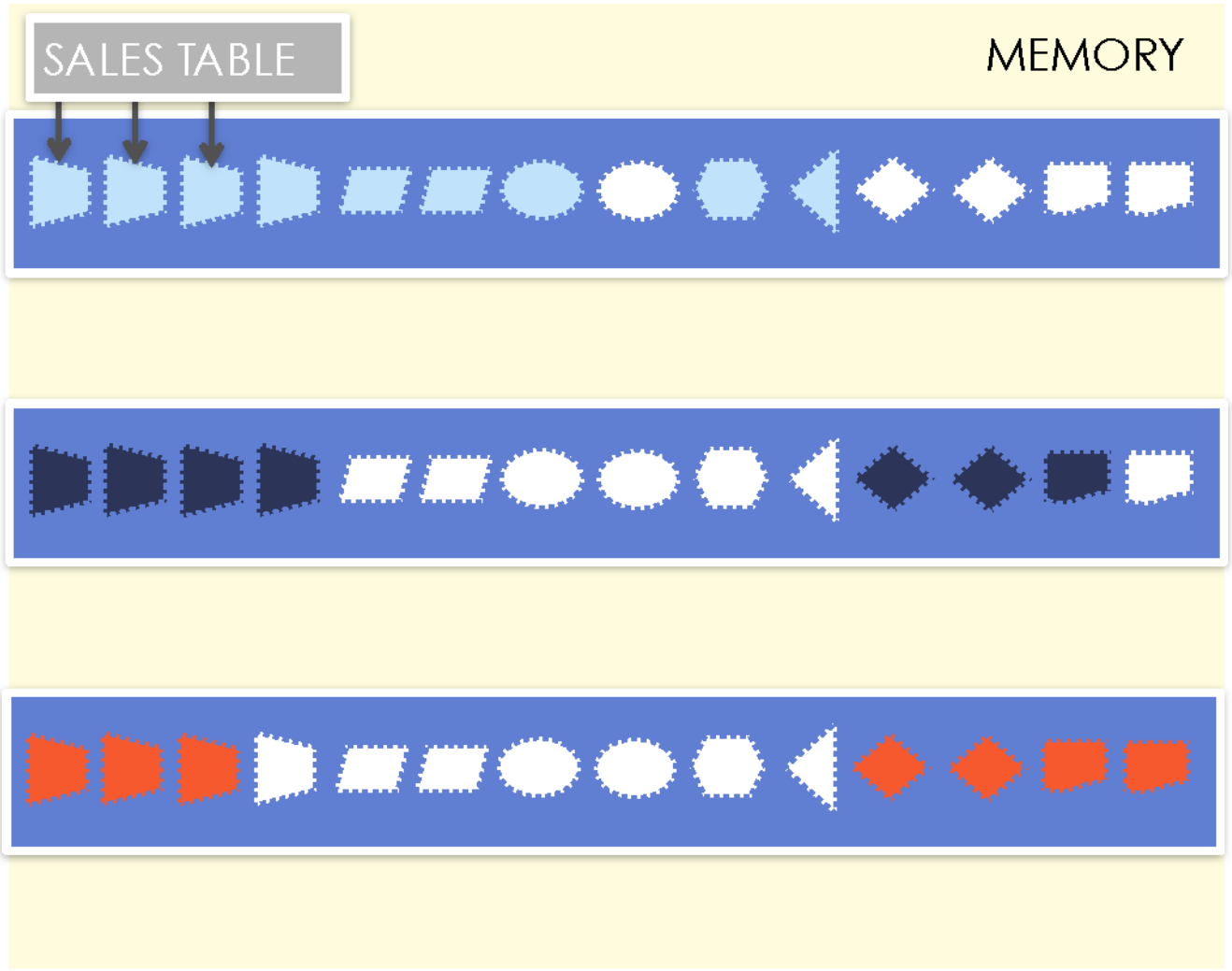
Temporal Instructions


Query
Plan
Q100 Program
SALES
Col1
Col2
Col3
Col4
Col5
Bool1
Table1
Table2
Table3
Table4
Table5
Col6
Col7
Col8
Col9
Col10
Col11
Col12
Col13
Table6
Table7
Final Answer
Spatial Instructions
Temporal Instructions

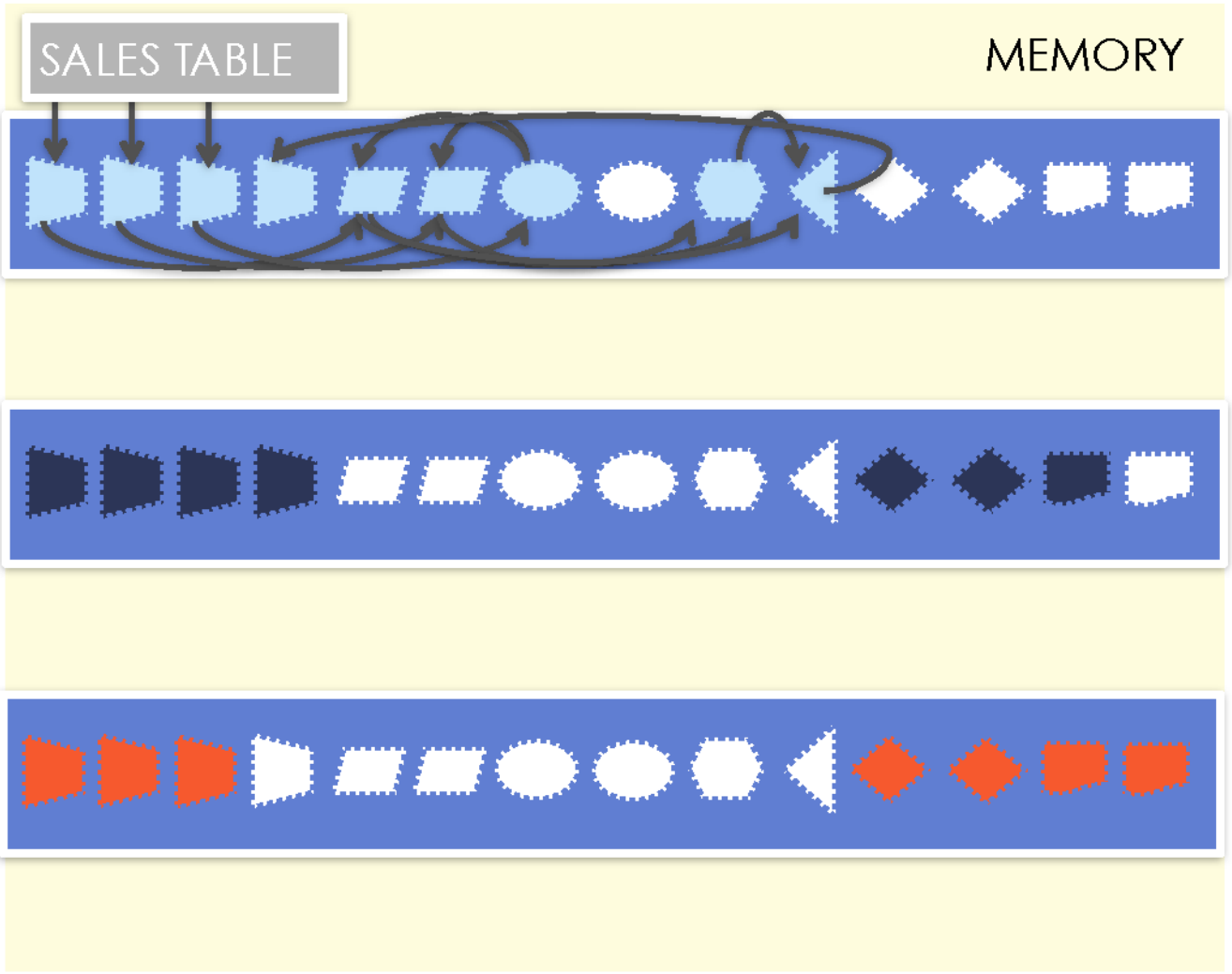
# Q100 Execution and Efficiencies



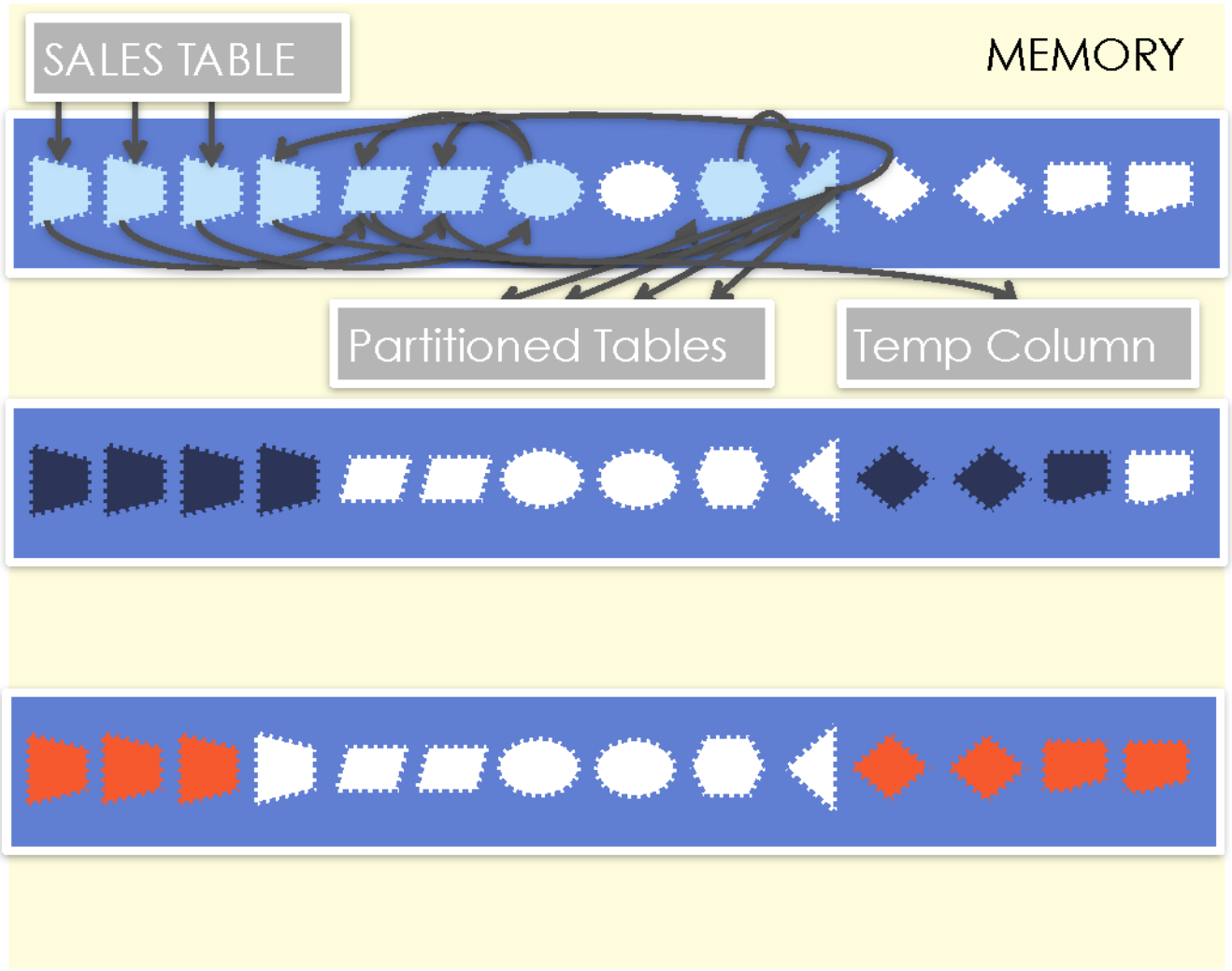
# Q100 Execution and Efficiencies



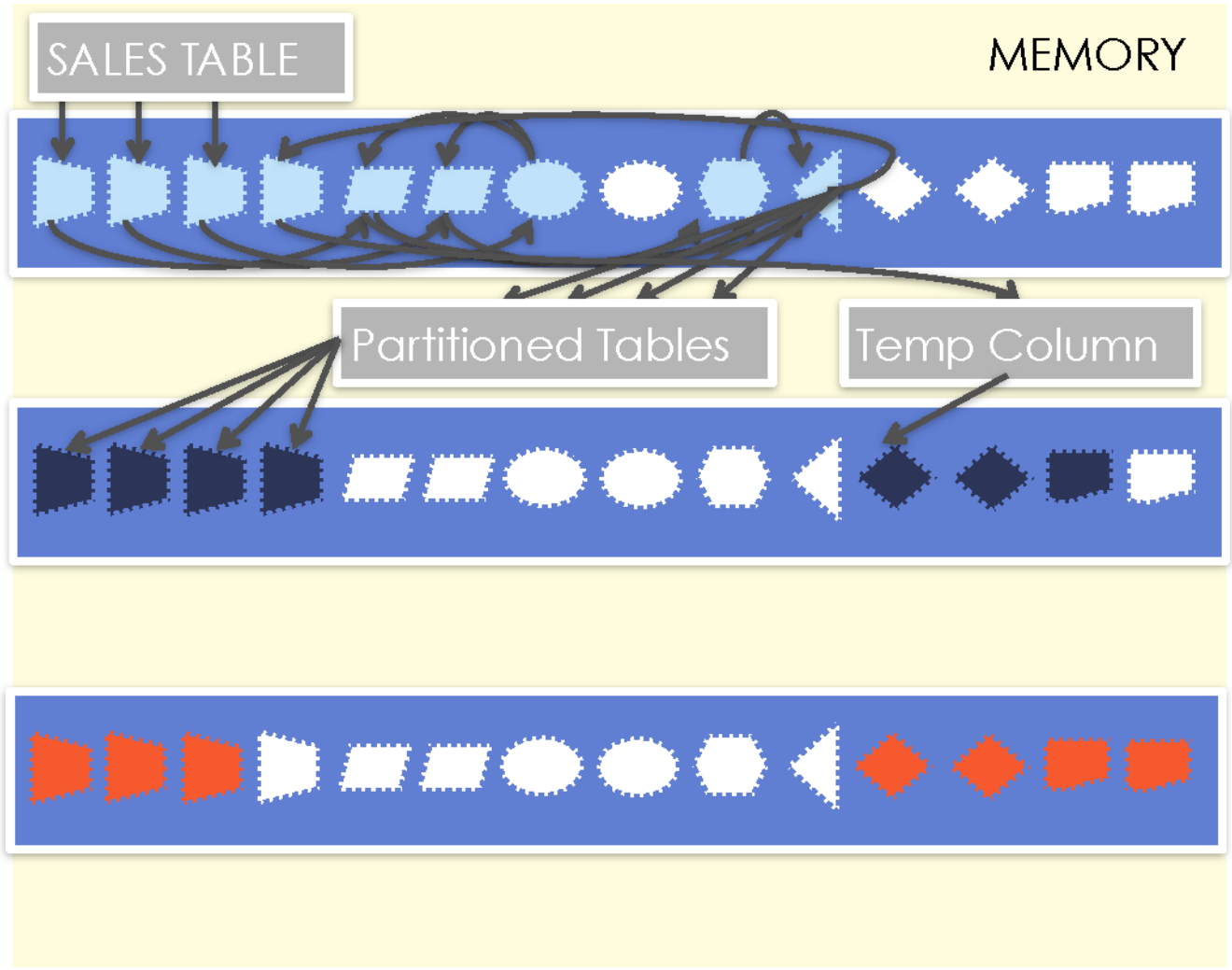
# Q100 Execution and Efficiencies



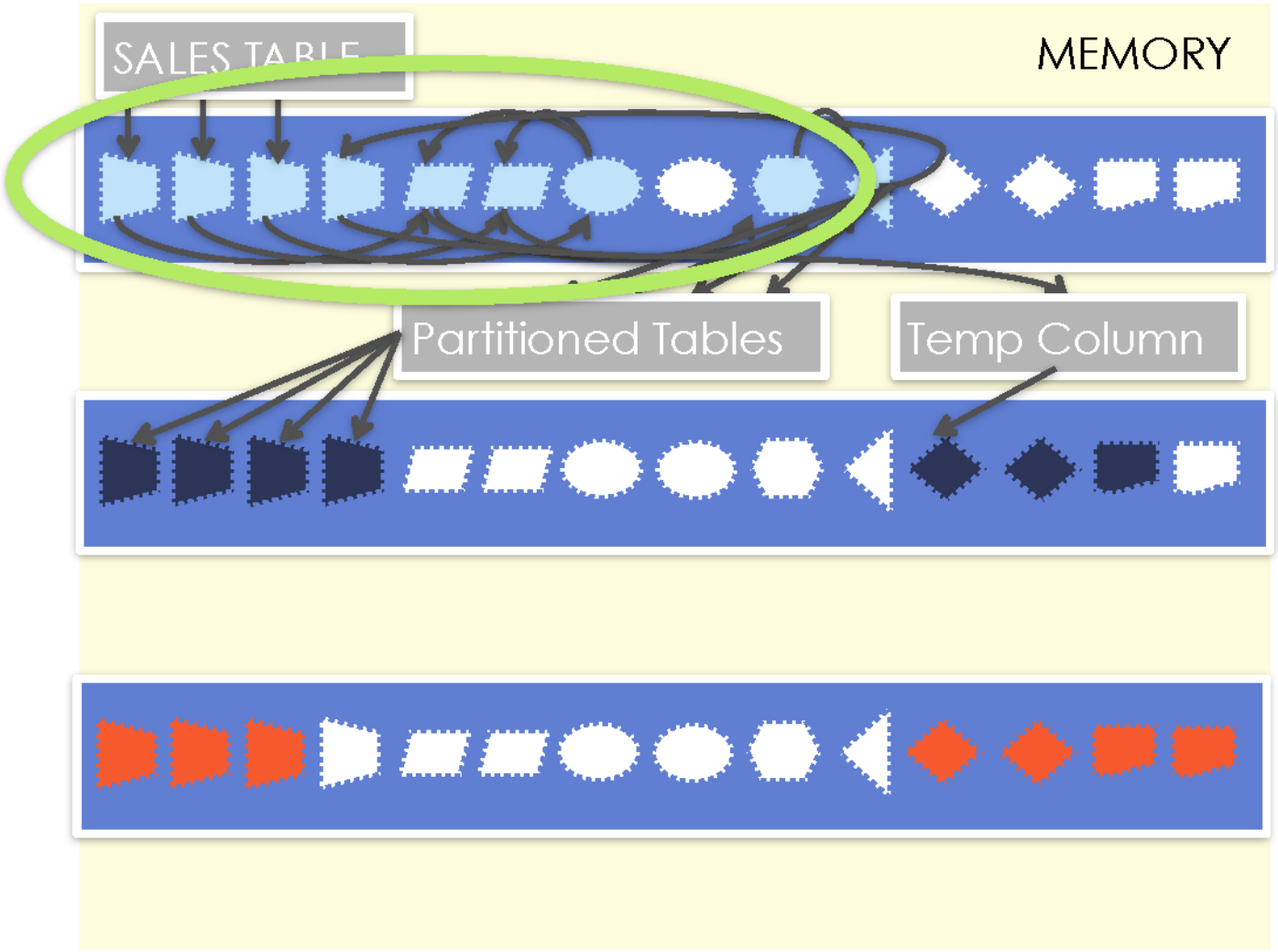
# Q100 Execution and Efficiencies



# Q100 Execution and Efficiencies

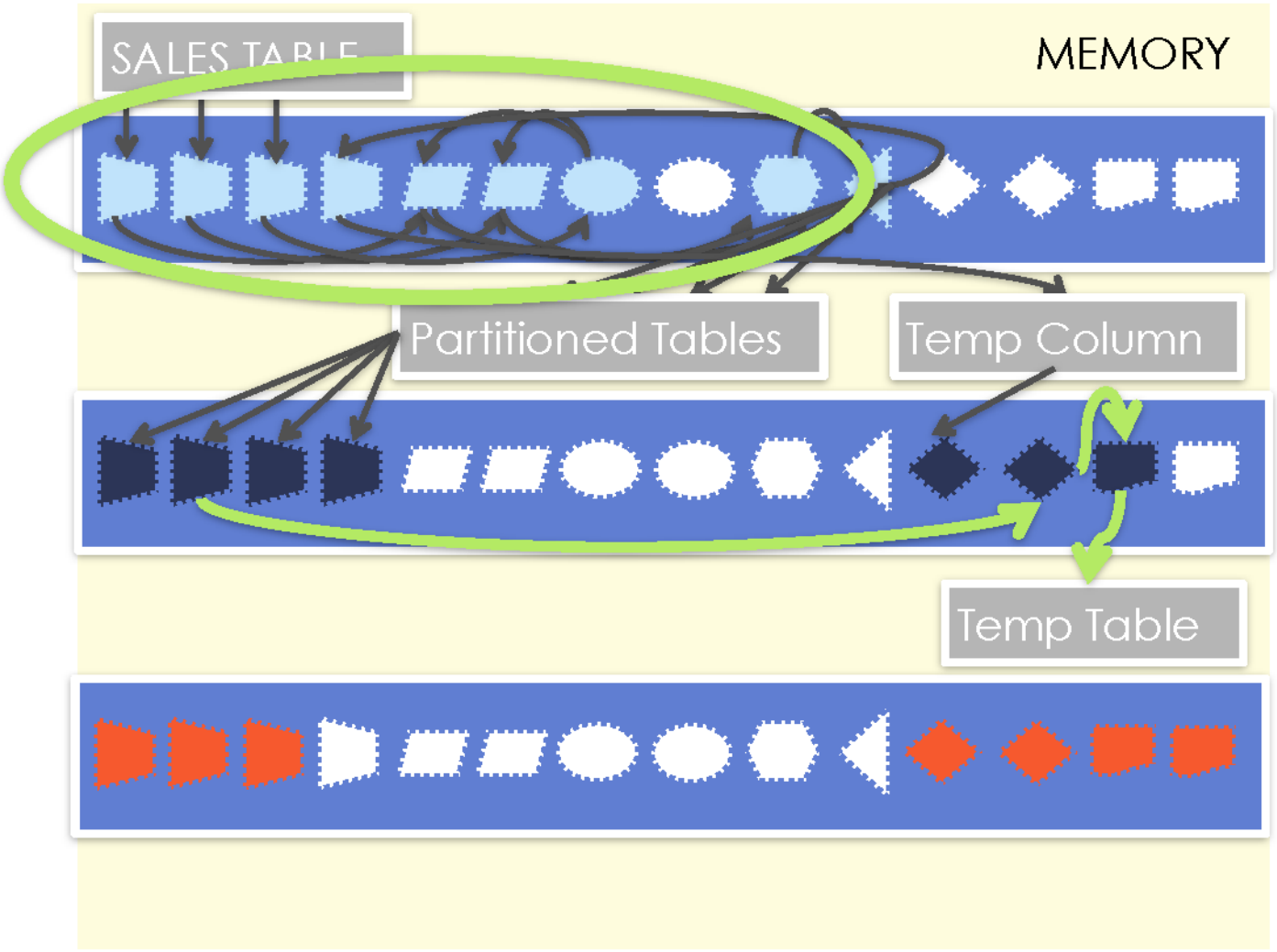


# Q100 Execution and Efficiencies




Read datum once, perform multiple operations

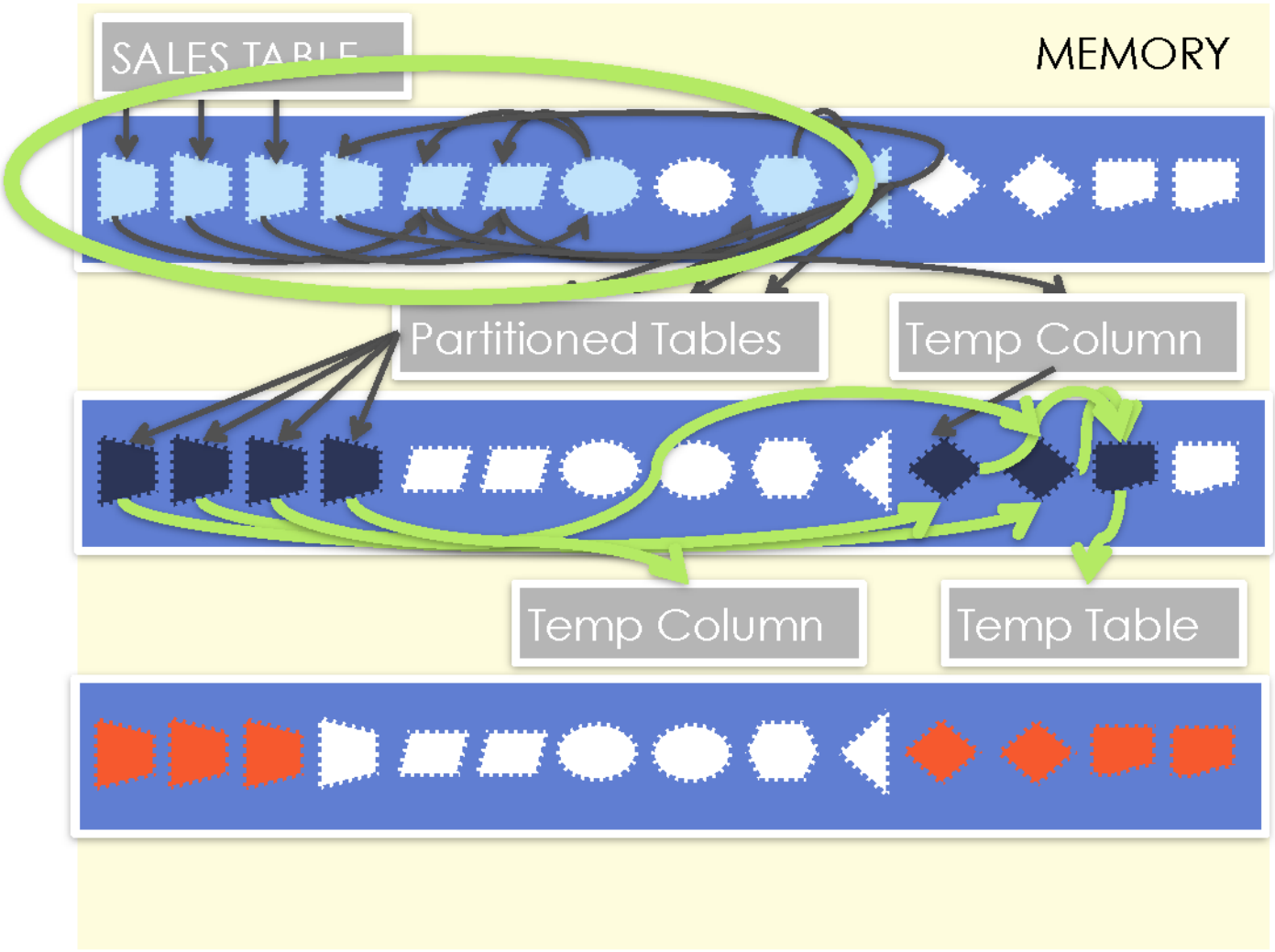
# Q100 Execution and Efficiencies




Read datum once, perform multiple operations

Pipeline Parallelism

# Q100 Execution and Efficiencies

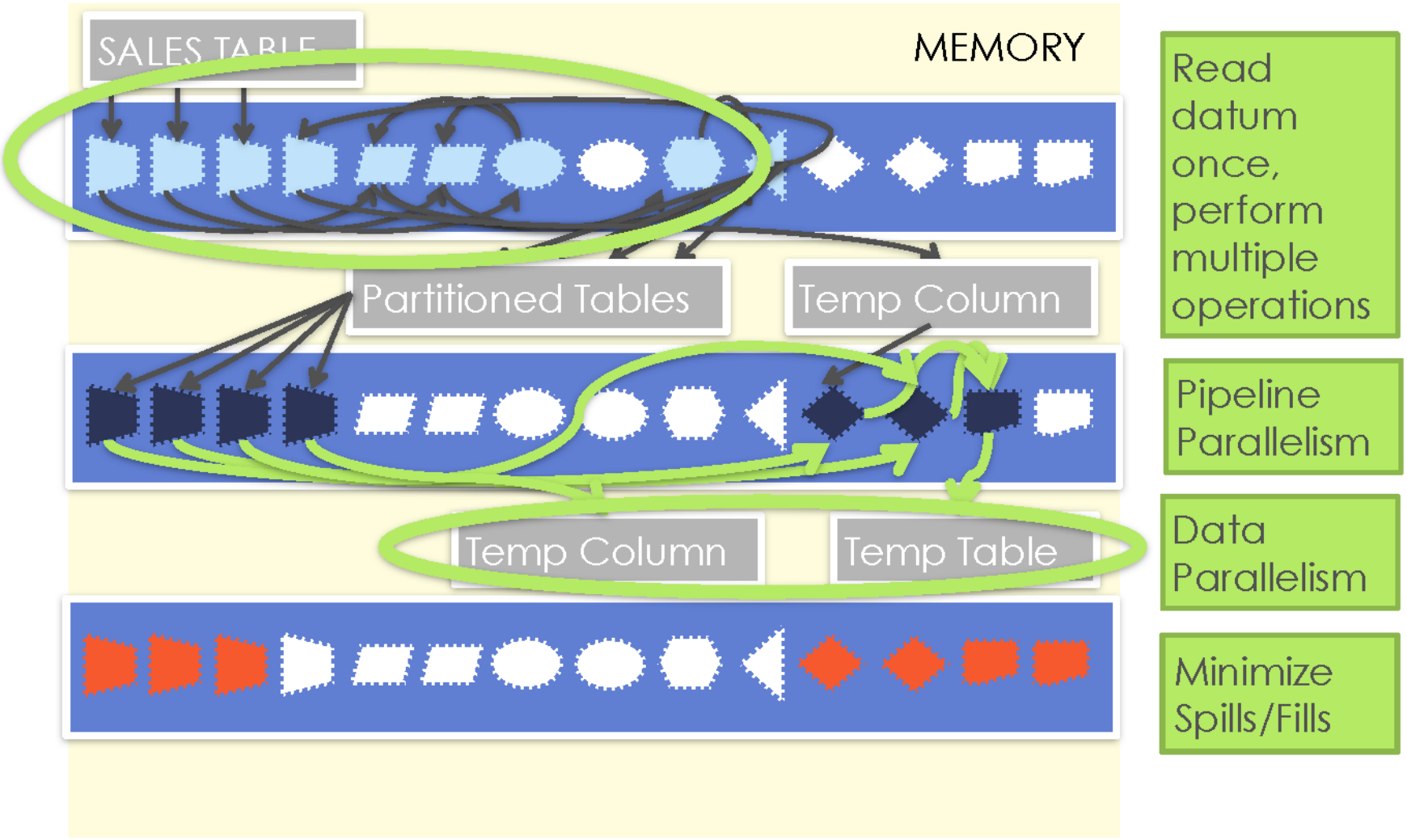



Read datum once, perform multiple operations

Pipeline Parallelism

Data Parallelism

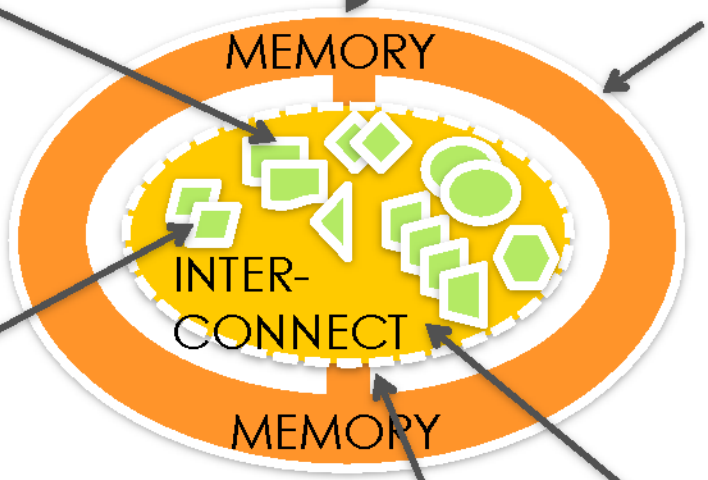
# Q100 Execution and Efficiencies



How do we generate these query plans?

How do we implement these operators?

How do we schedule the plans?




How many tiles should there be and of what type?

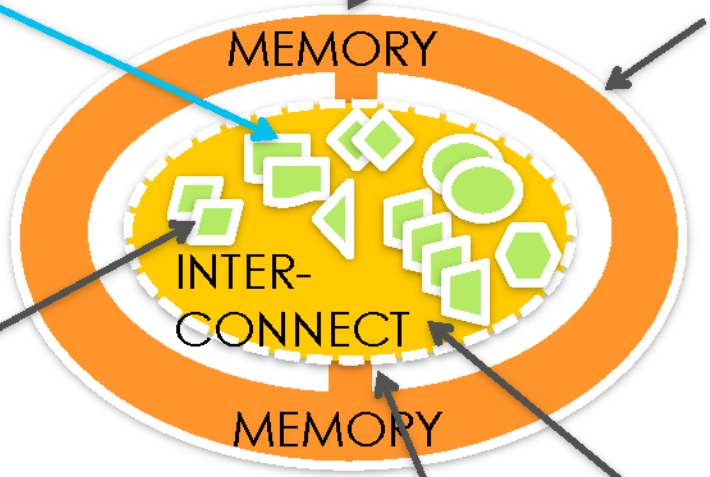
Is the Q100 performance and energy efficient?

What kind of interconnect should we use?

How do we generate these query plans?

How do we implement these operators?

How do we schedule the plans?




How many tiles should there be and of what type?

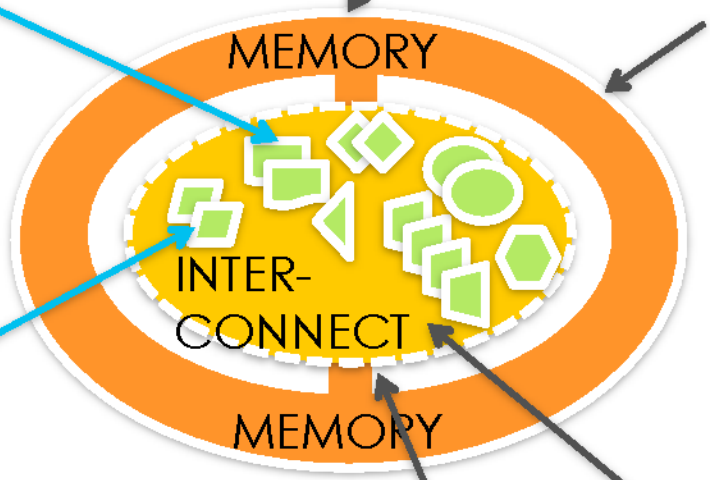
Is the Q100 performance and energy efficient?

What kind of interconnect should we use?

How do we generate these query plans?

How do we schedule the plans?

How do we implement these operators?




How many tiles should there be and of what type?

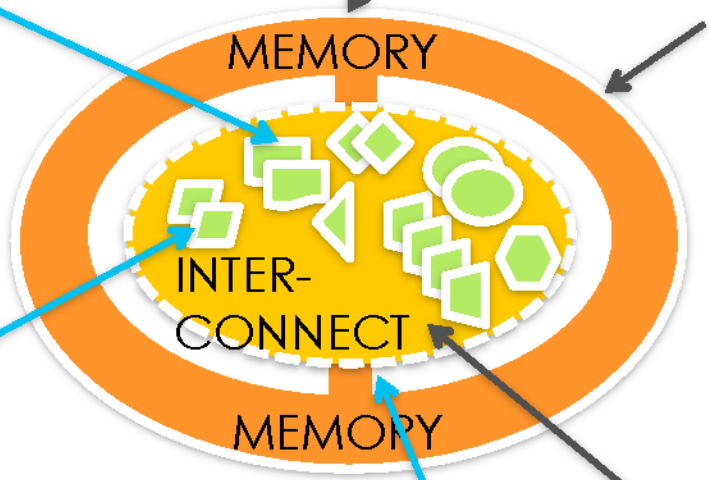
Is the Q100 performance and energy efficient?

What kind of interconnect should we use?

How do we generate these query plans?

How do we schedule the plans?

How do we implement these operators?




How many tiles should there be and of what type?

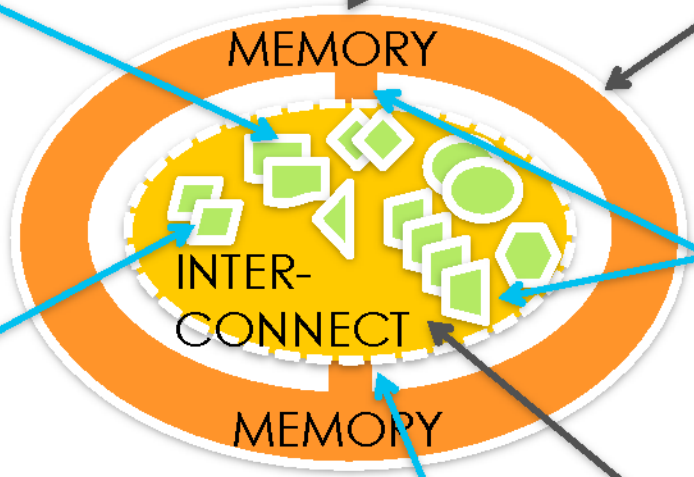
Is the Q100 performance and energy efficient?

What kind of interconnect should we use?

How do we generate these query plans?

How do we implement these operators?

How do we schedule the plans?




Bandwidth needs on- and off- chip

How many tiles should there be and of what type?

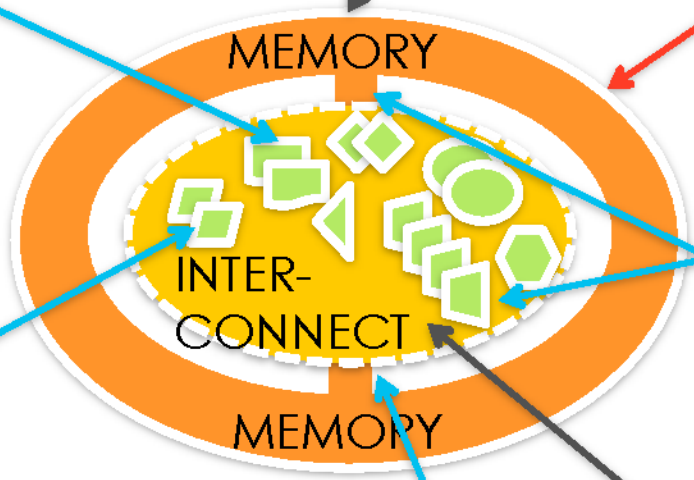
Is the Q100 performance and energy efficient?

What kind of interconnect should we use?

How do we generate these query plans?

How do we implement these operators?

How do we schedule the plans?



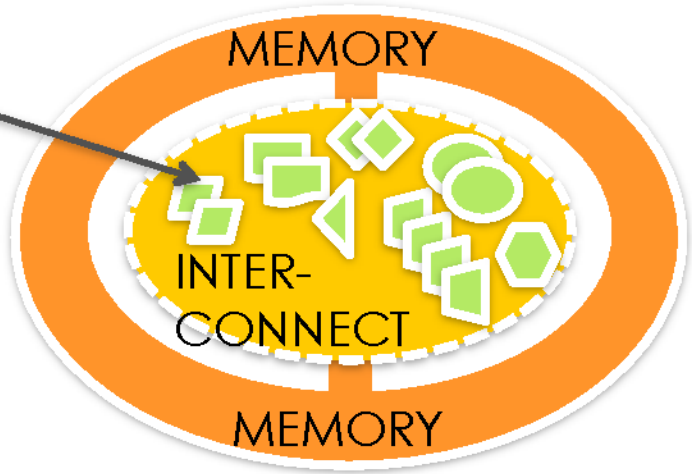

Bandwidth needs on- and off- chip

How many tiles should there be and of what type?

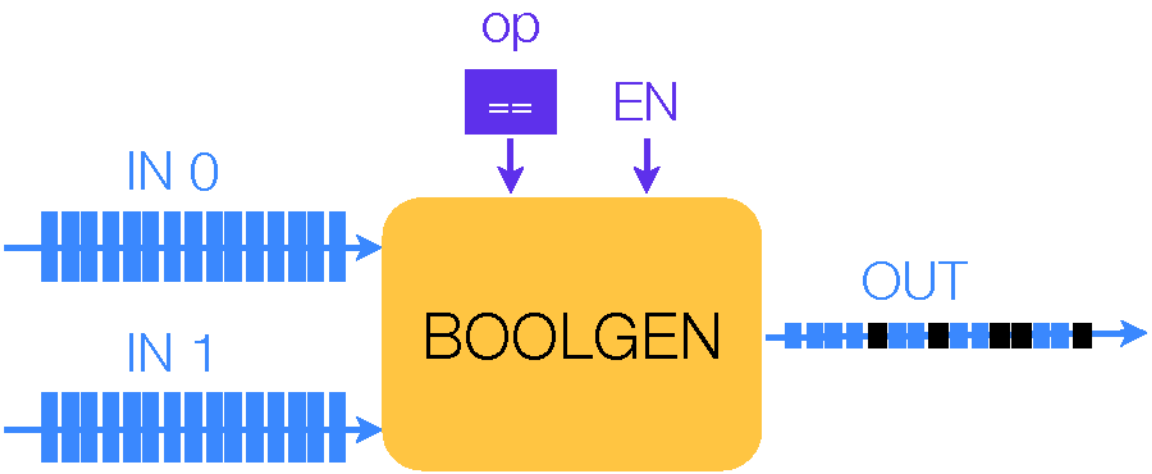
Is the Q100 performance and energy efficient?

What kind of interconnect should we use?

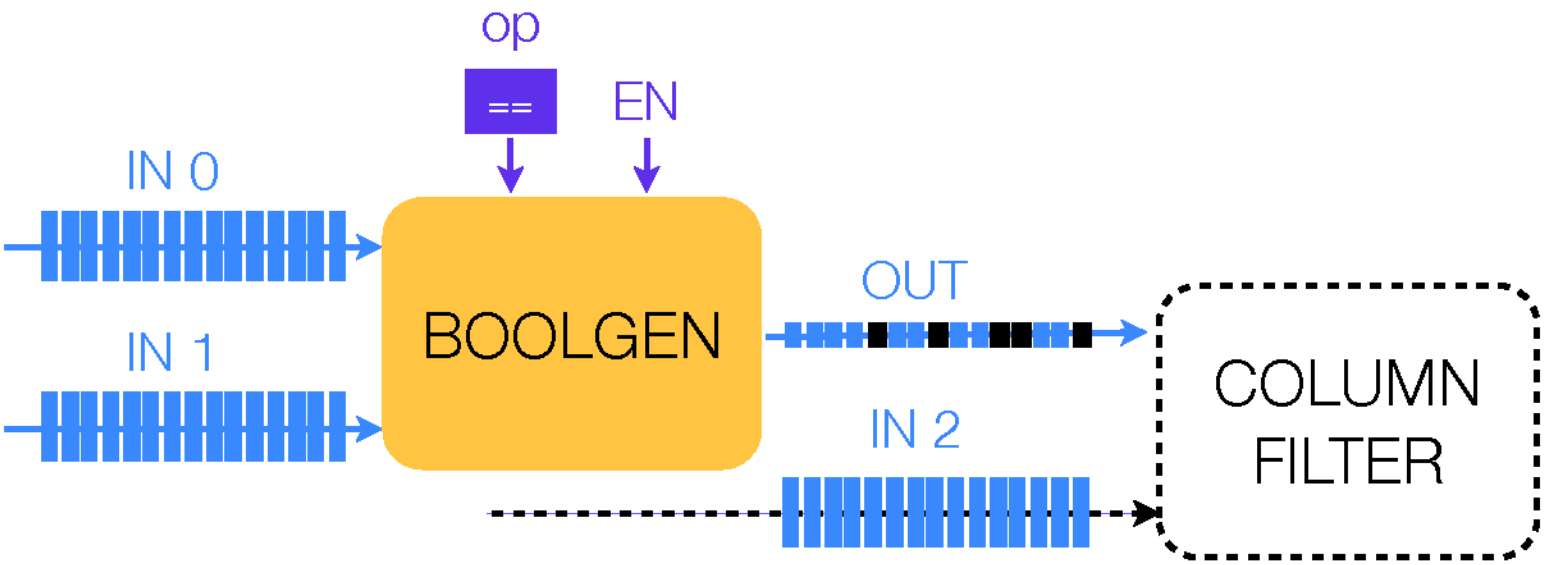
How do we implement these operators?



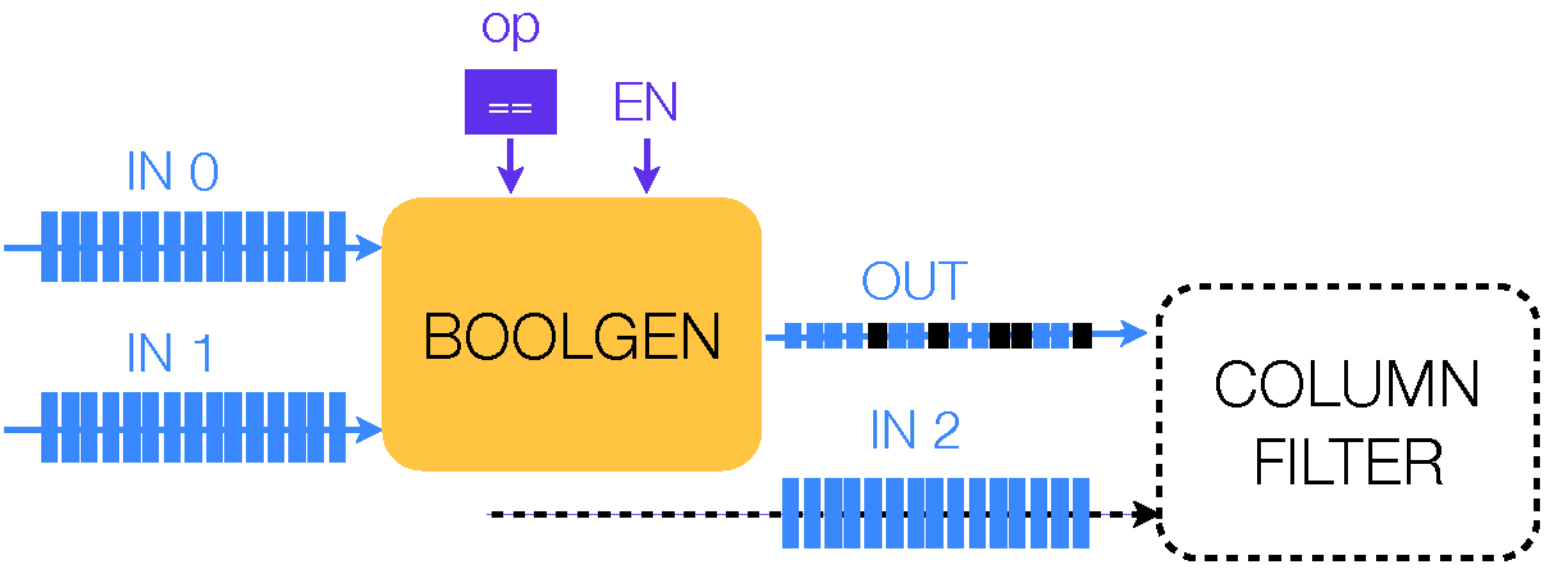
# Example Tile: Boolean Generator



# Example Tile: Boolean Generator

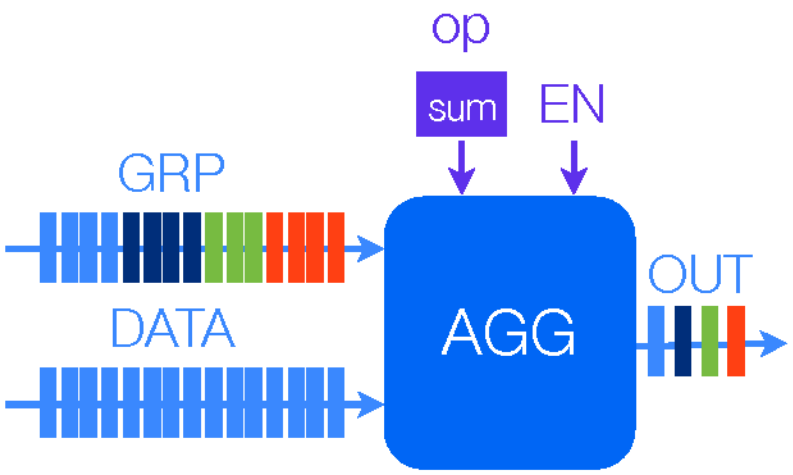


# Example Tile: Boolean Generator

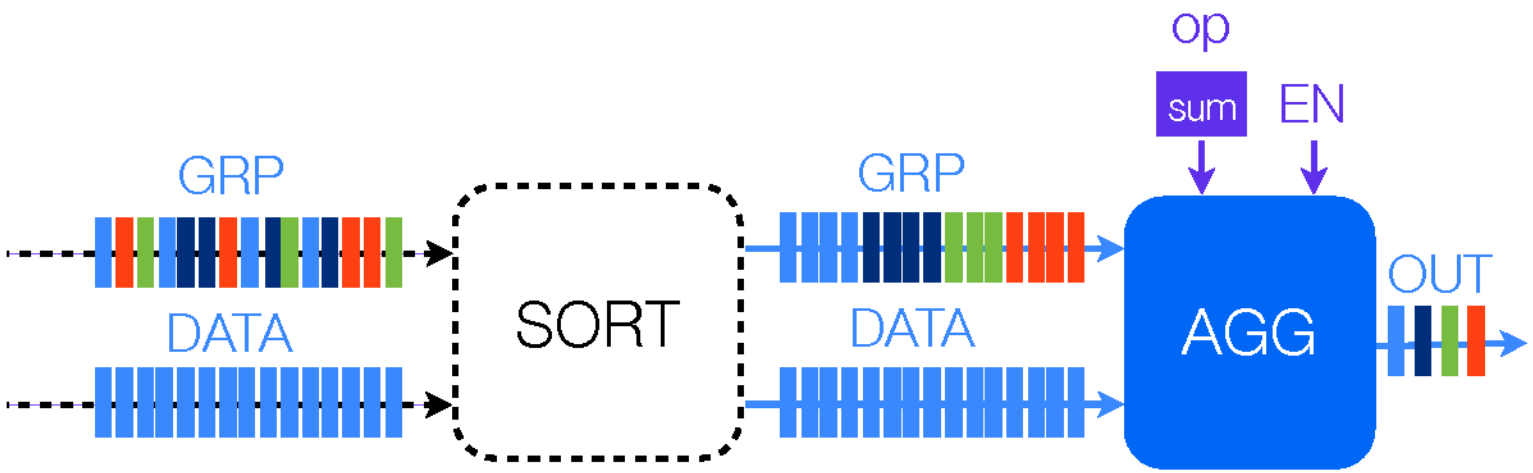



WHERE s_shipdate >= '2013-01-01'

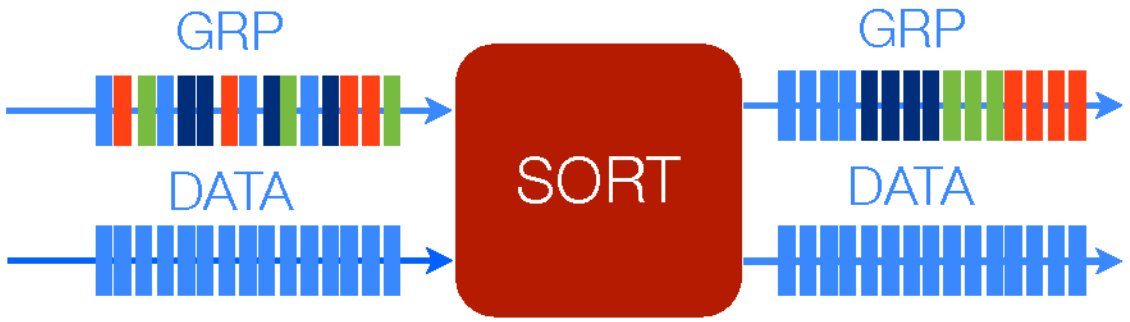
# Example Tile: Aggregator



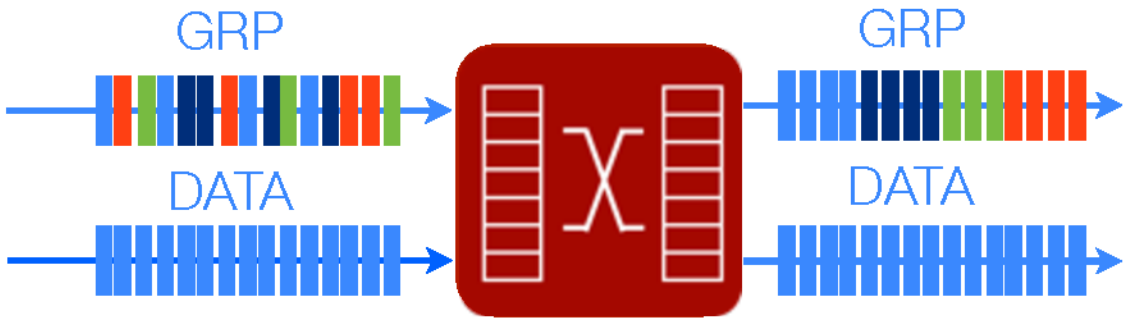
# Example Tile: Aggregator



# Example Tile: Sorter

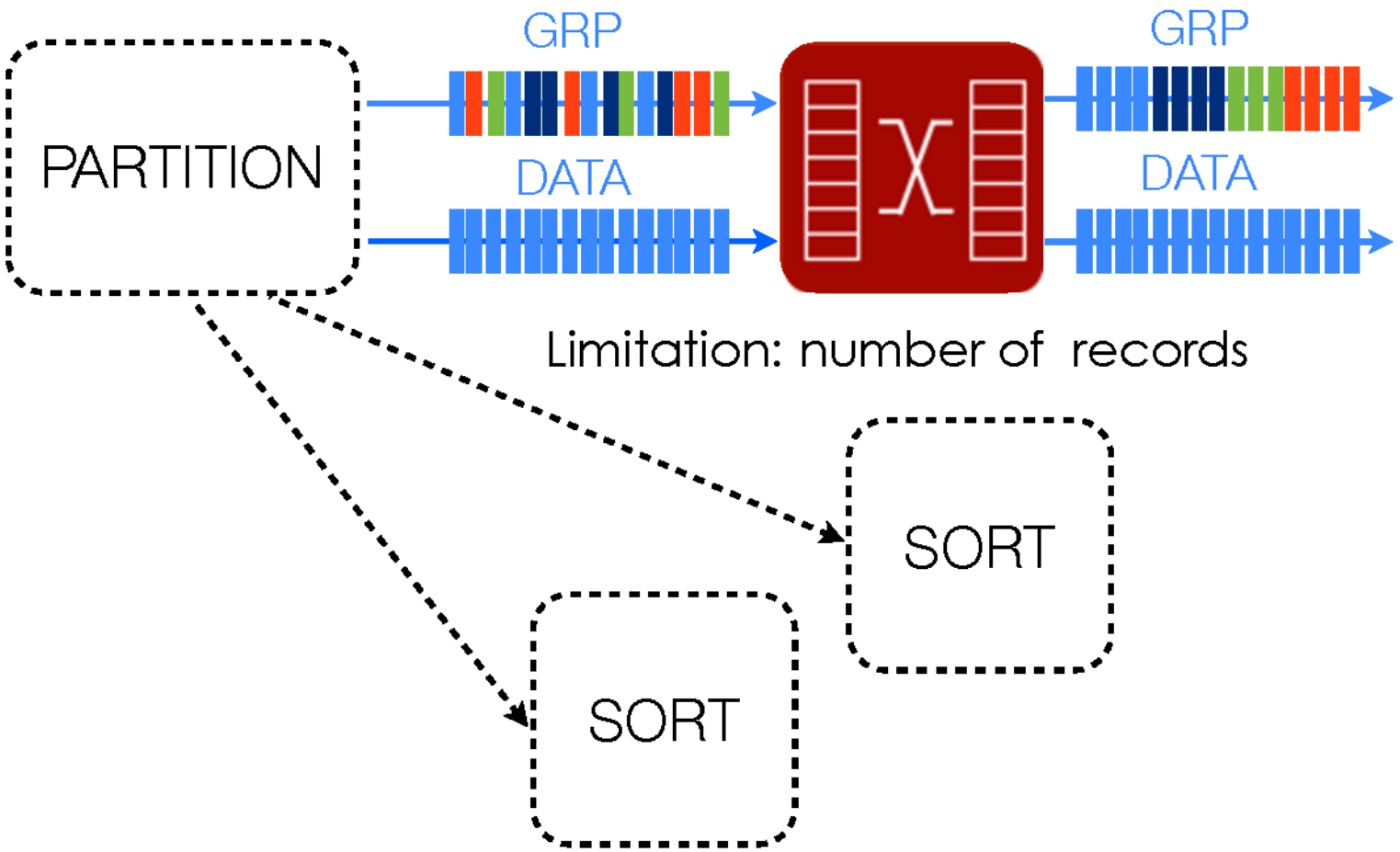


# Example Tile: Sorter




Limitation: number of records

# Example Tile: Sorter



# Q100 Tiles

- **Functional Tiles (7)**
  Aggregator
  ALU
  Boolean Generator
  Column Filter
  Joiner
  Partitioner
  Sorter

# Q100 Tiles

- **Functional Tiles (7)**
  Aggregator
  ALU
  Boolean Generator
  Column Filter
  Joiner
  Partitioner
  Sorter
- **Auxiliary Tiles (4)**
  Table Appender
  Column Selector
  Column Concatenator
  Column Stitcher

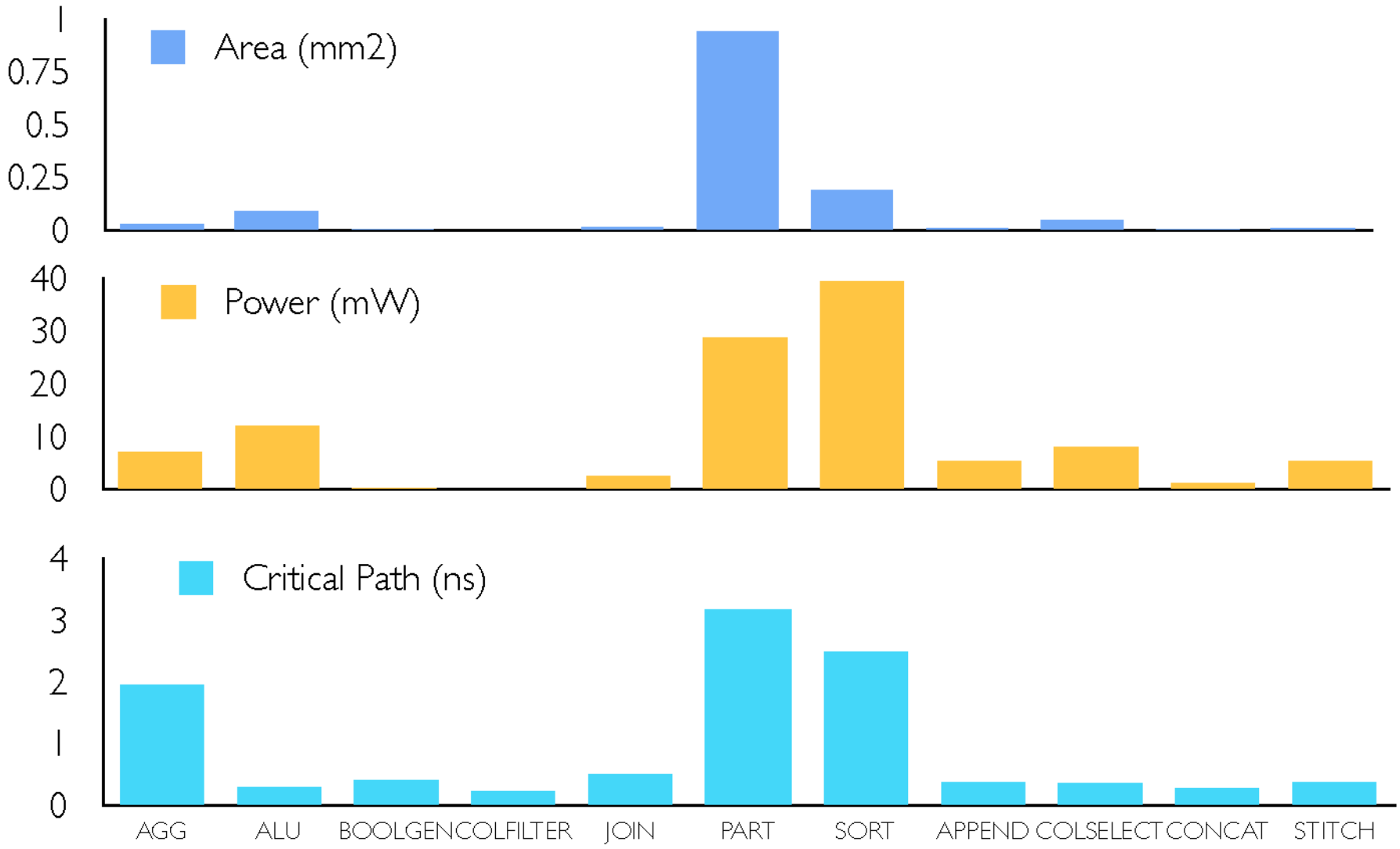
# Q100 Tiles

- **Functional Tiles (7)**
  - Aggregator
  - ALU
  - Boolean Generator
  - Column Filter
  - Joiner
  - Partitioner
  - Sorter
- **Auxiliary Tiles (4)**
  - Table Appender
  - Column Selector
  - Column Concatenator
  - Column Stitcher

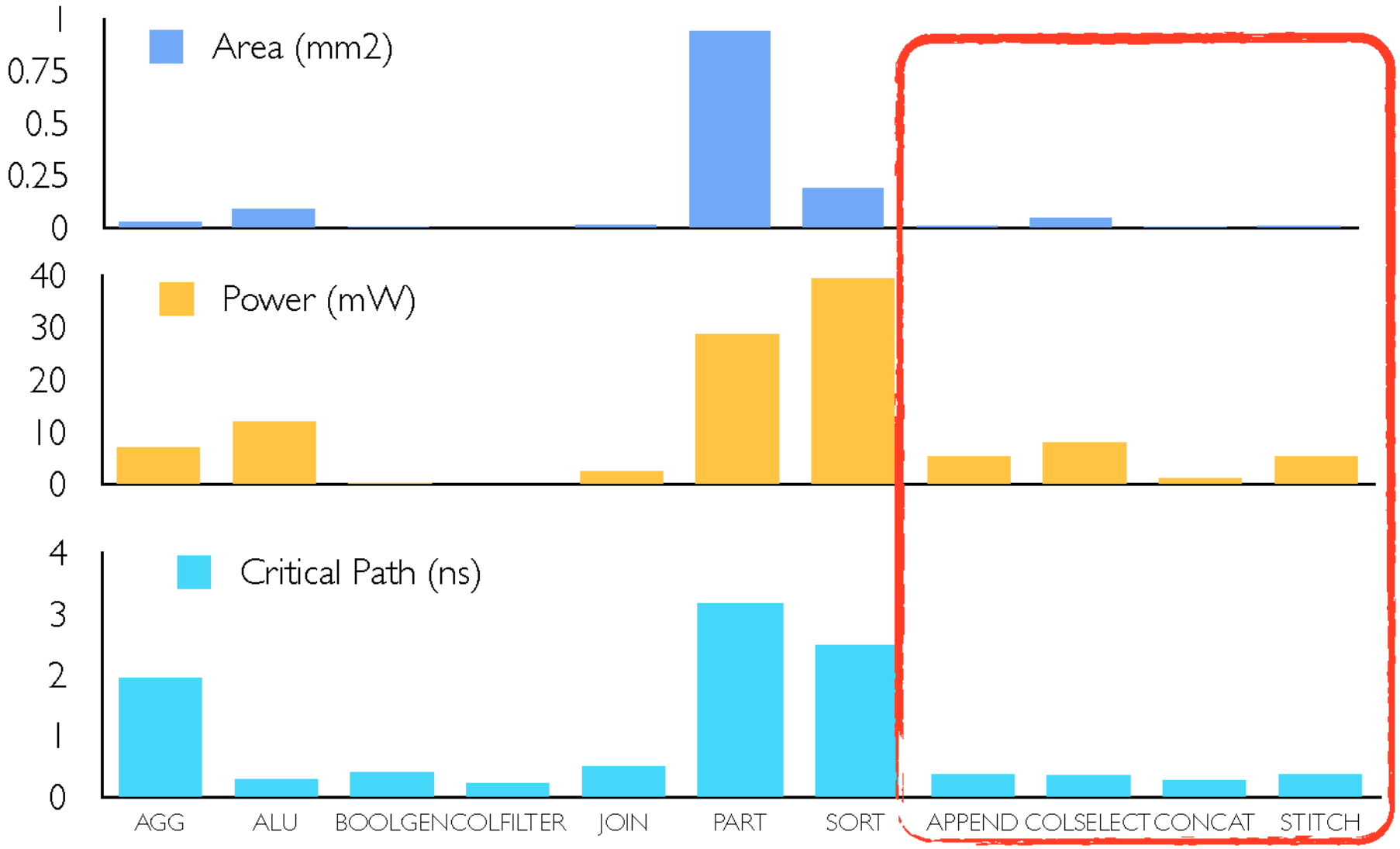
**Tile Characterization Methodology**

Verilog implementation for each tile, synthesized, placed, and routed using Synopsys 32nm Generic Libraries

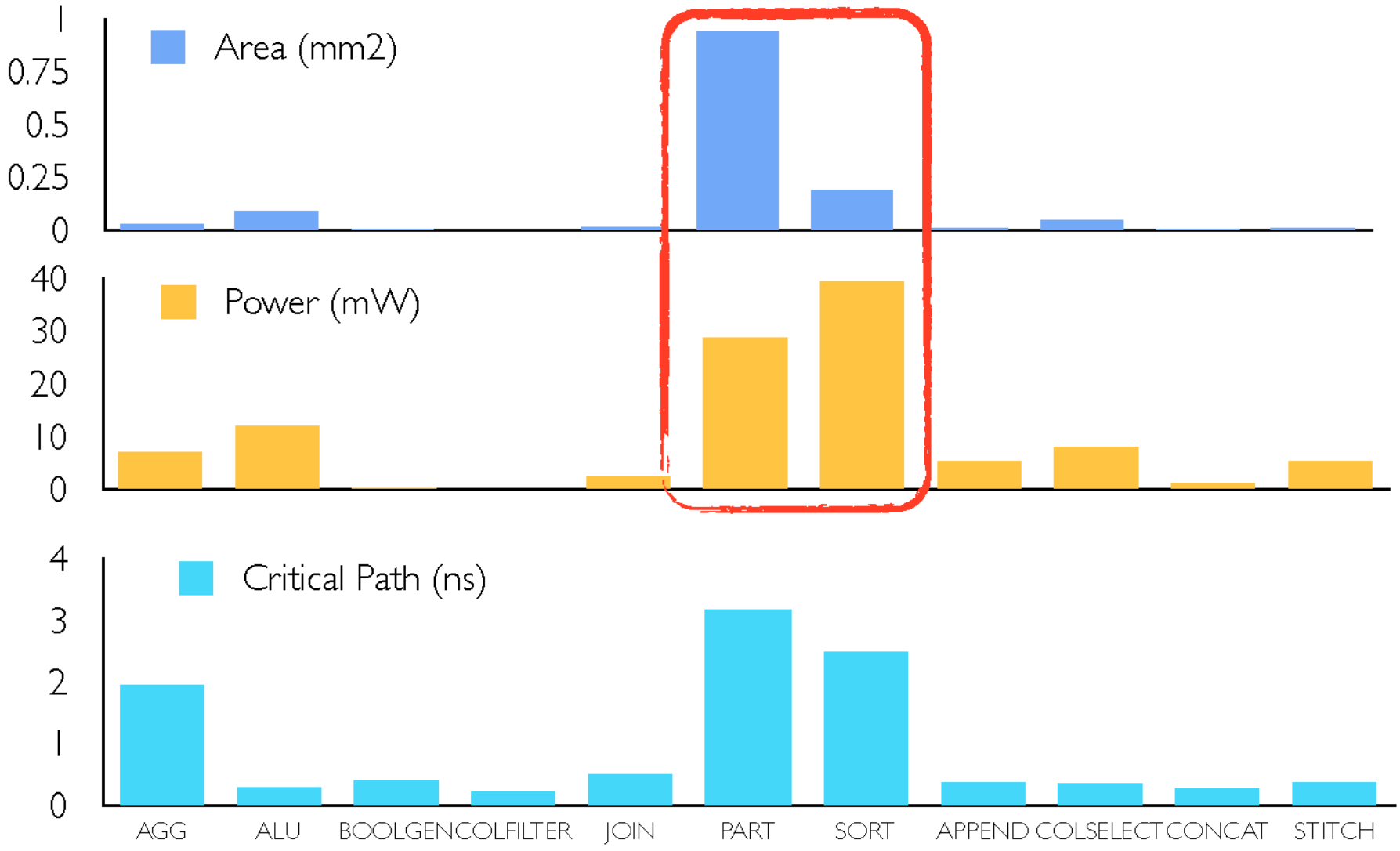
# Tile Characterization



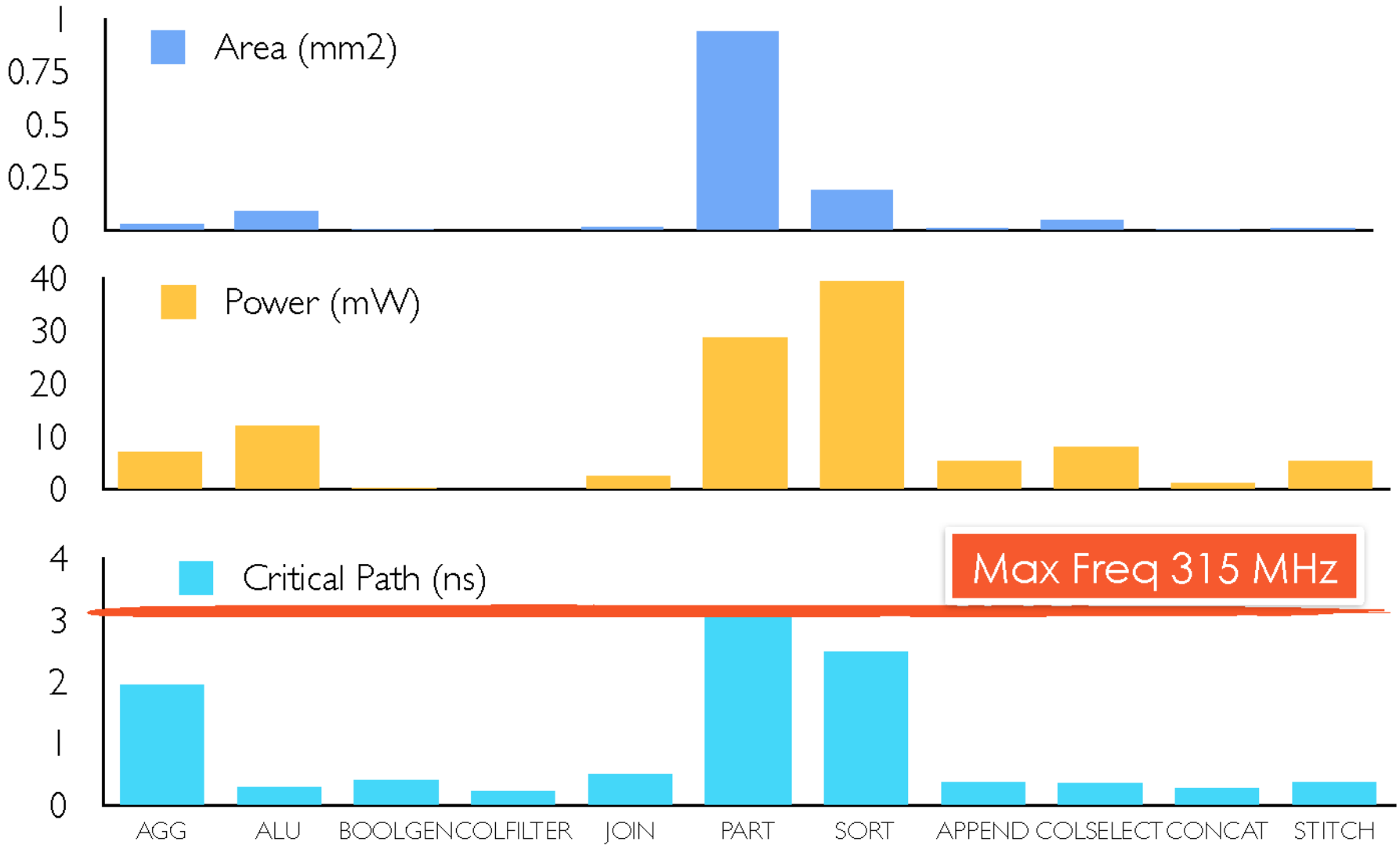
# Tile Characterization

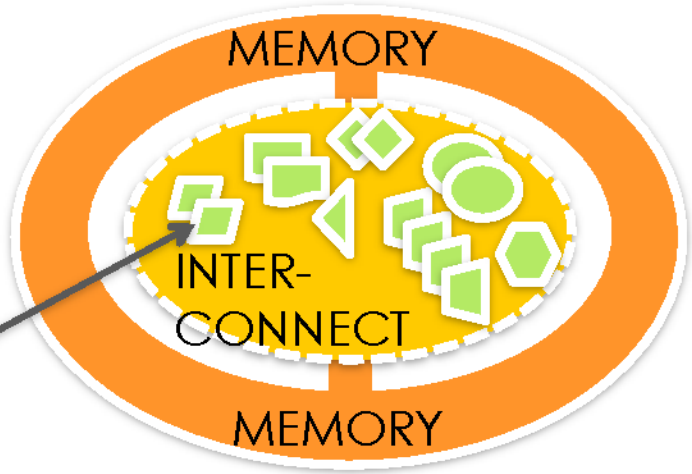


# Tile Characterization



# Tile Characterization






How many tiles should there be and of what type?

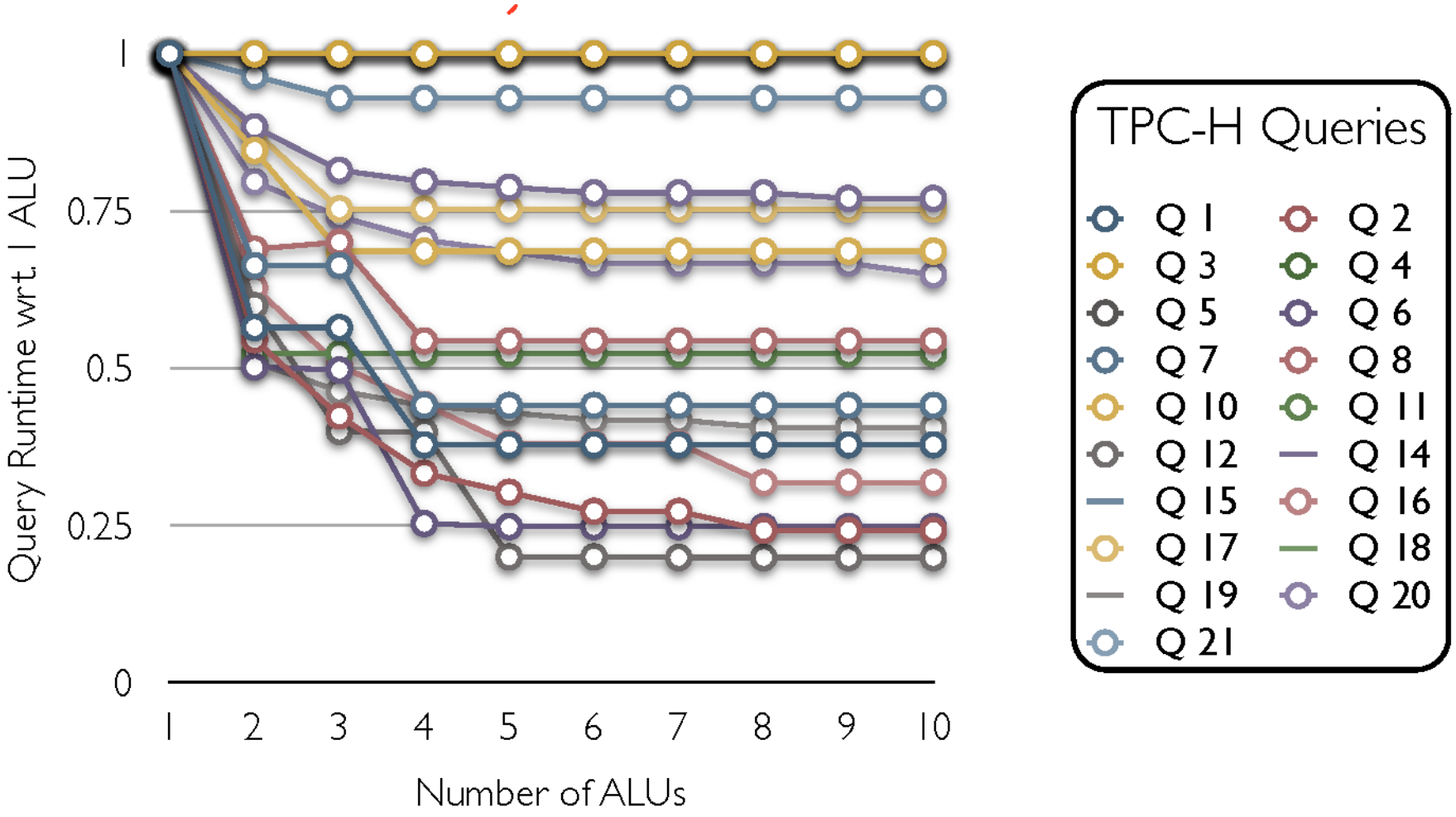
# Unbounded Design Space

| | |
|---|---|
| AGGREGATOR | 1 2 3 4 5 6 7 8 9 10 11 12... |
| ALU | 1 2 3 4 5 6 7 8 9 10 11 12... |
| BOOLEAN GENERATOR | 1 2 3 4 5 6 7 8 9 10 11 12... |
| COLUMN FILTER | 1 2 3 4 5 6 7 8 9 10 11 12... |
| JOINER | 1 2 3 4 5 6 7 8 9 10 11 12... |
| PARTITIONER | 1 2 3 4 5 6 7 8 9 10 11 12... |
| SORTER | 1 2 3 4 5 6 7 8 9 10 11 12... |
| TABLE APPENDER | 1 2 3 4 5 6 7 8 9 10 11 12... |
| COLUMN SELECTOR | 1 2 3 4 5 6 7 8 9 10 11 12... |
| COLUMN CONCATENATOR | 1 2 3 4 5 6 7 8 9 10 11 12... |
| COLUMN STITCHER | 1 2 3 4 5 6 7 8 9 10 11 12... |

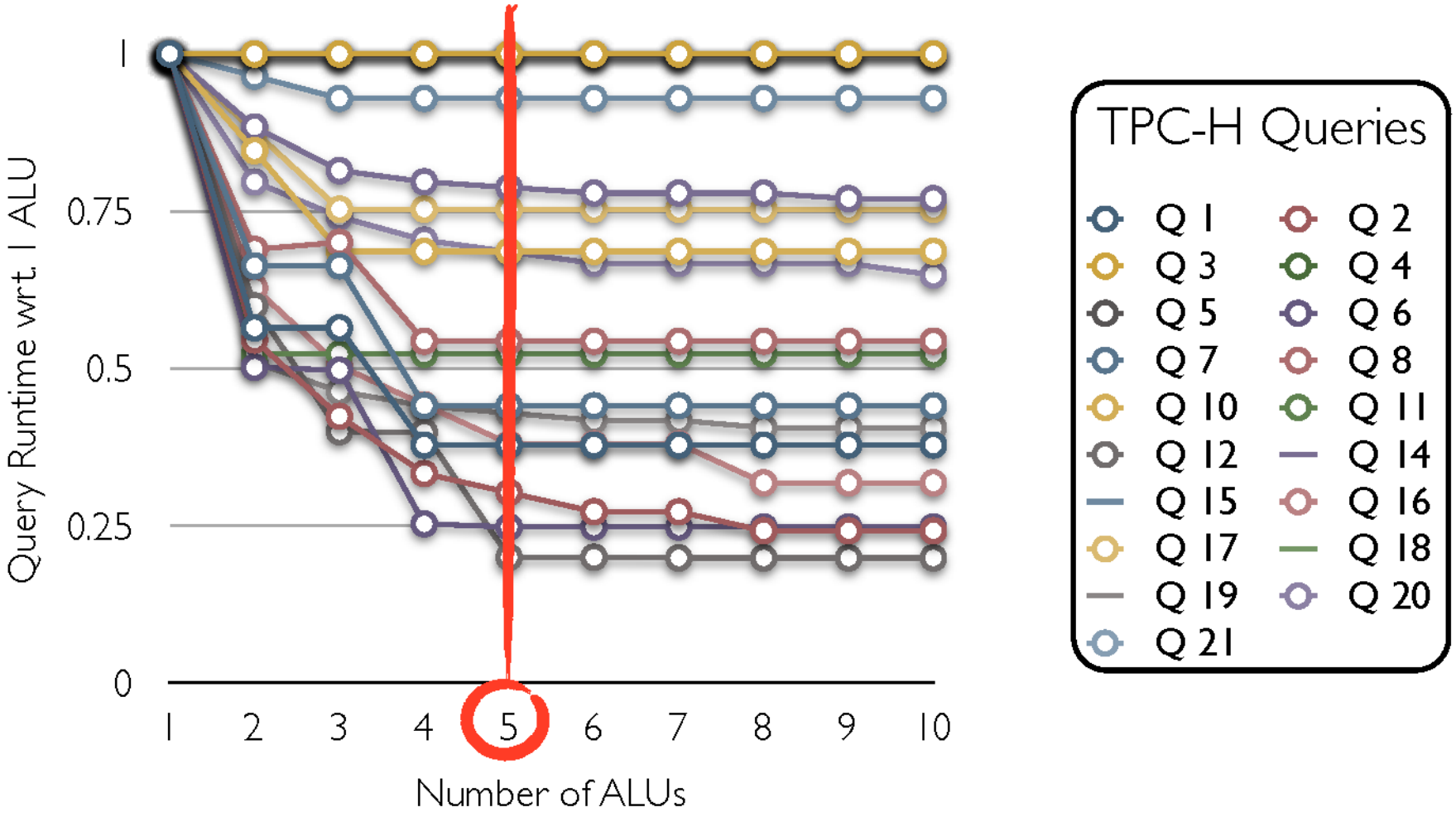
# Performance Simulation Methodology

- TPC-H as target workload
- Home-grown C++ simulator, validated against MonetDB
- Completion cycles for each spatial and temporal instructions
- Memory access overheads
- Completion time for each query converted to throughput using the Q100 frequency

# Example: Bounding ALU Count



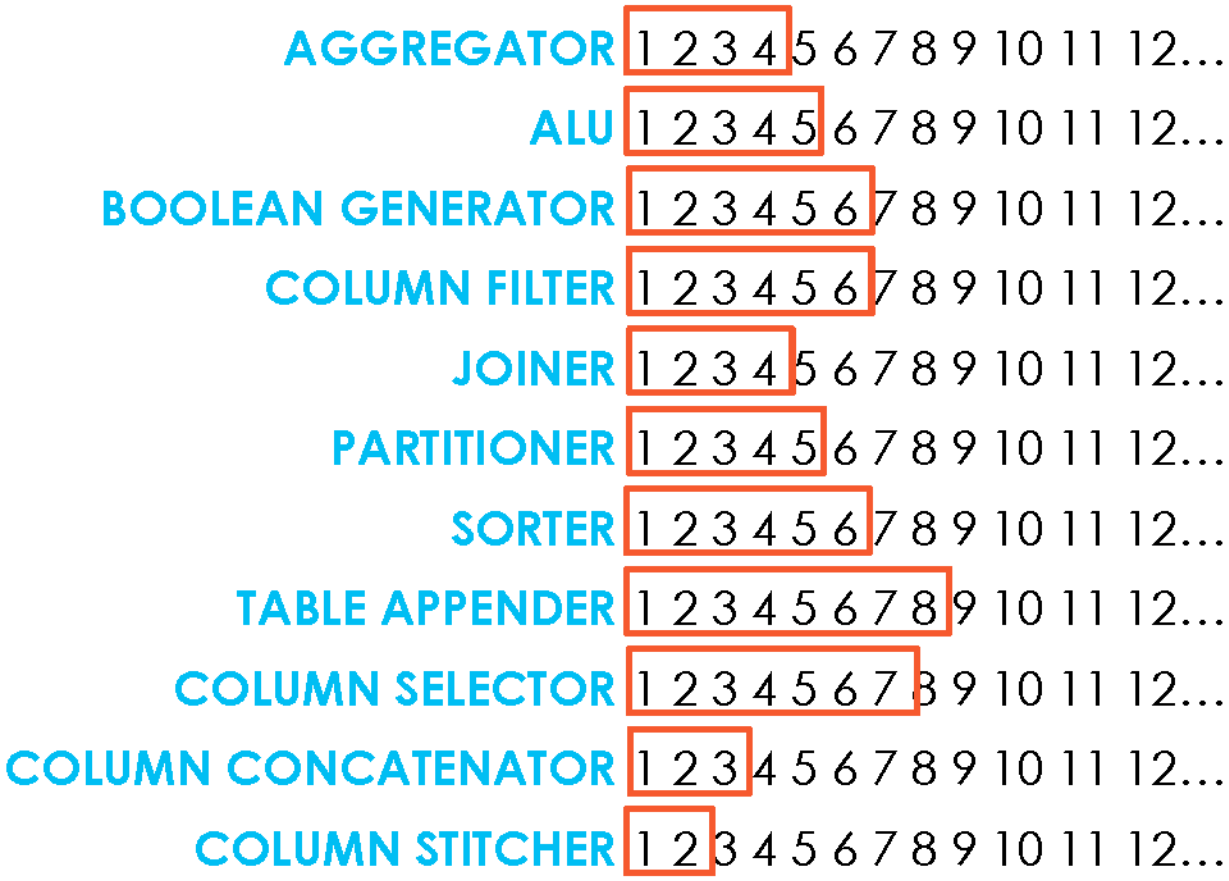
# Example: Bounding ALU Count



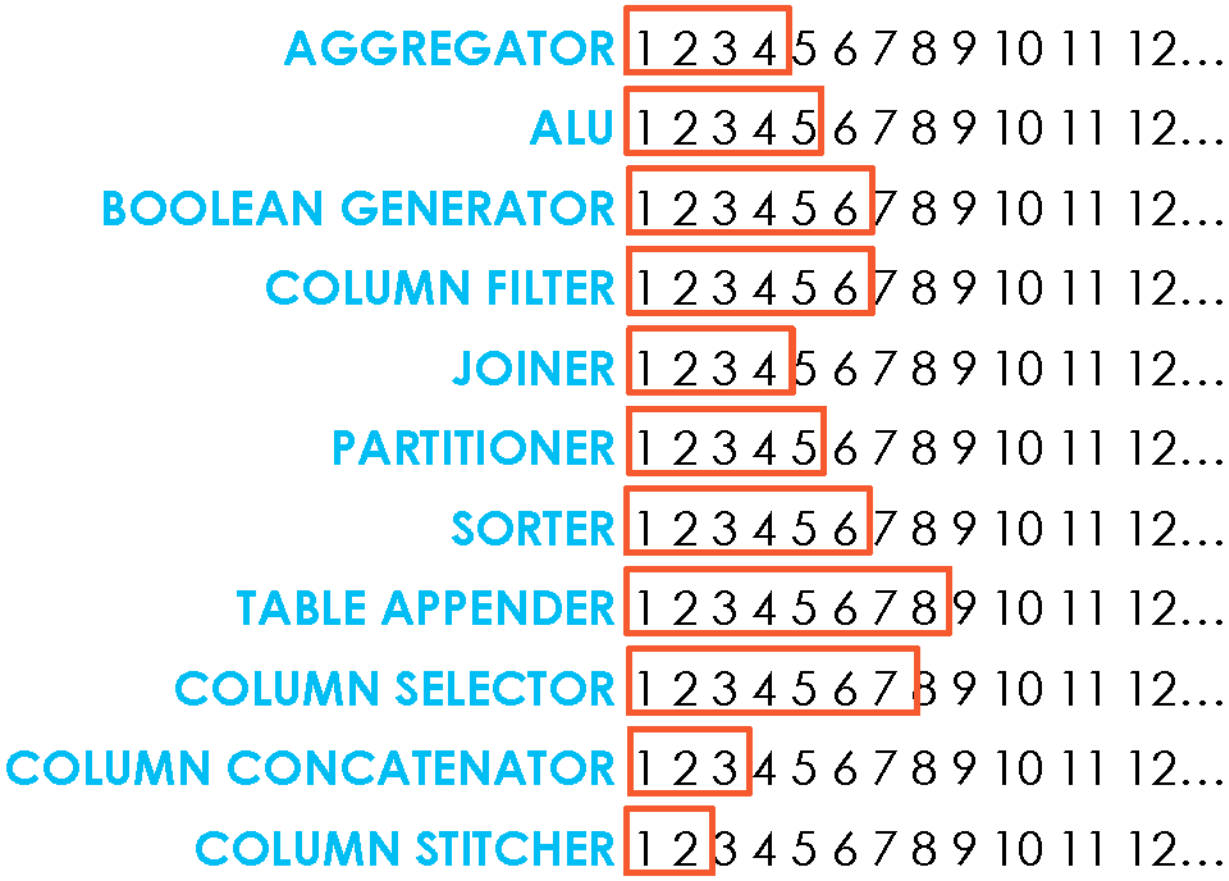
# Bounded Design Space

| | |
|---|---|
| AGGREGATOR | 1 2 3 4 5 6 7 8 9 10 11 12... |
| ALU | 1 2 3 4 5 6 7 8 9 10 11 12... |
| BOOLEAN GENERATOR | 1 2 3 4 5 6 7 8 9 10 11 12... |
| COLUMN FILTER | 1 2 3 4 5 6 7 8 9 10 11 12... |
| JOINER | 1 2 3 4 5 6 7 8 9 10 11 12... |
| PARTITIONER | 1 2 3 4 5 6 7 8 9 10 11 12... |
| SORTER | 1 2 3 4 5 6 7 8 9 10 11 12... |
| TABLE APPENDER | 1 2 3 4 5 6 7 8 9 10 11 12... |
| COLUMN SELECTOR | 1 2 3 4 5 6 7 8 9 10 11 12... |
| COLUMN CONCATENATOR | 1 2 3 4 5 6 7 8 9 10 11 12... |
| COLUMN STITCHER | 1 2 3 4 5 6 7 8 9 10 11 12... |

# Bounded Design Space

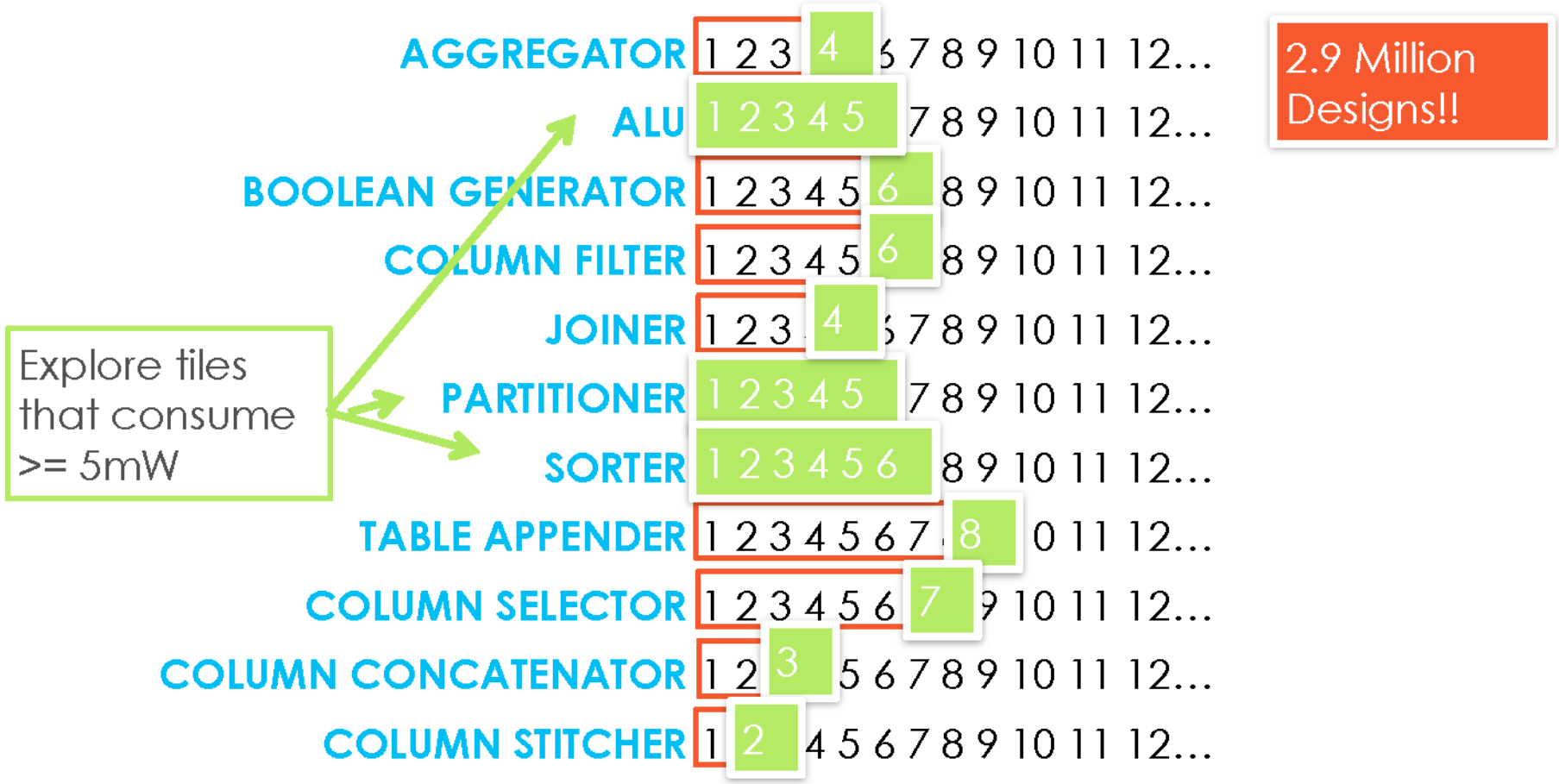


# Bounded Design Space

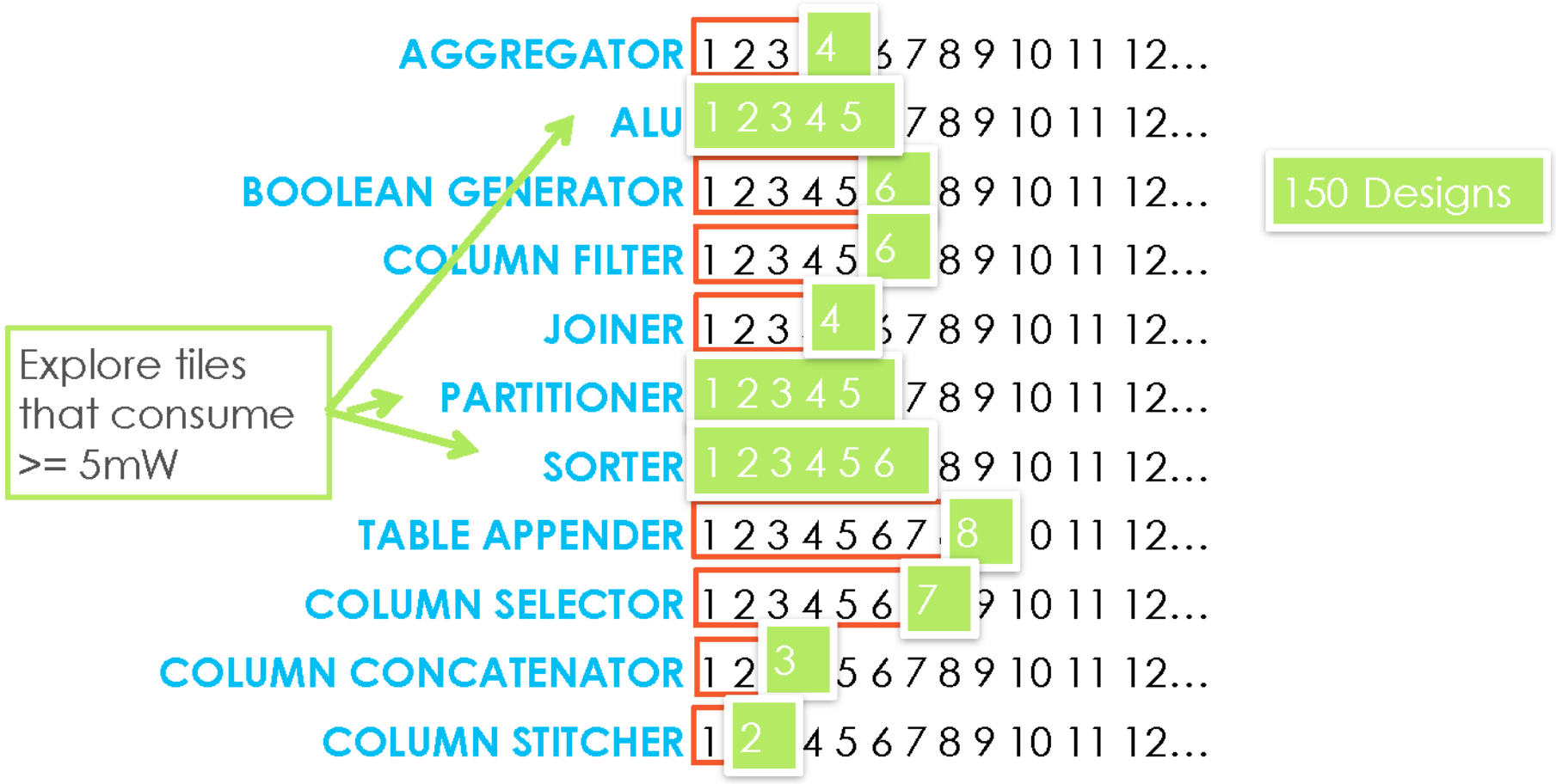



2.9 Million Designs!!

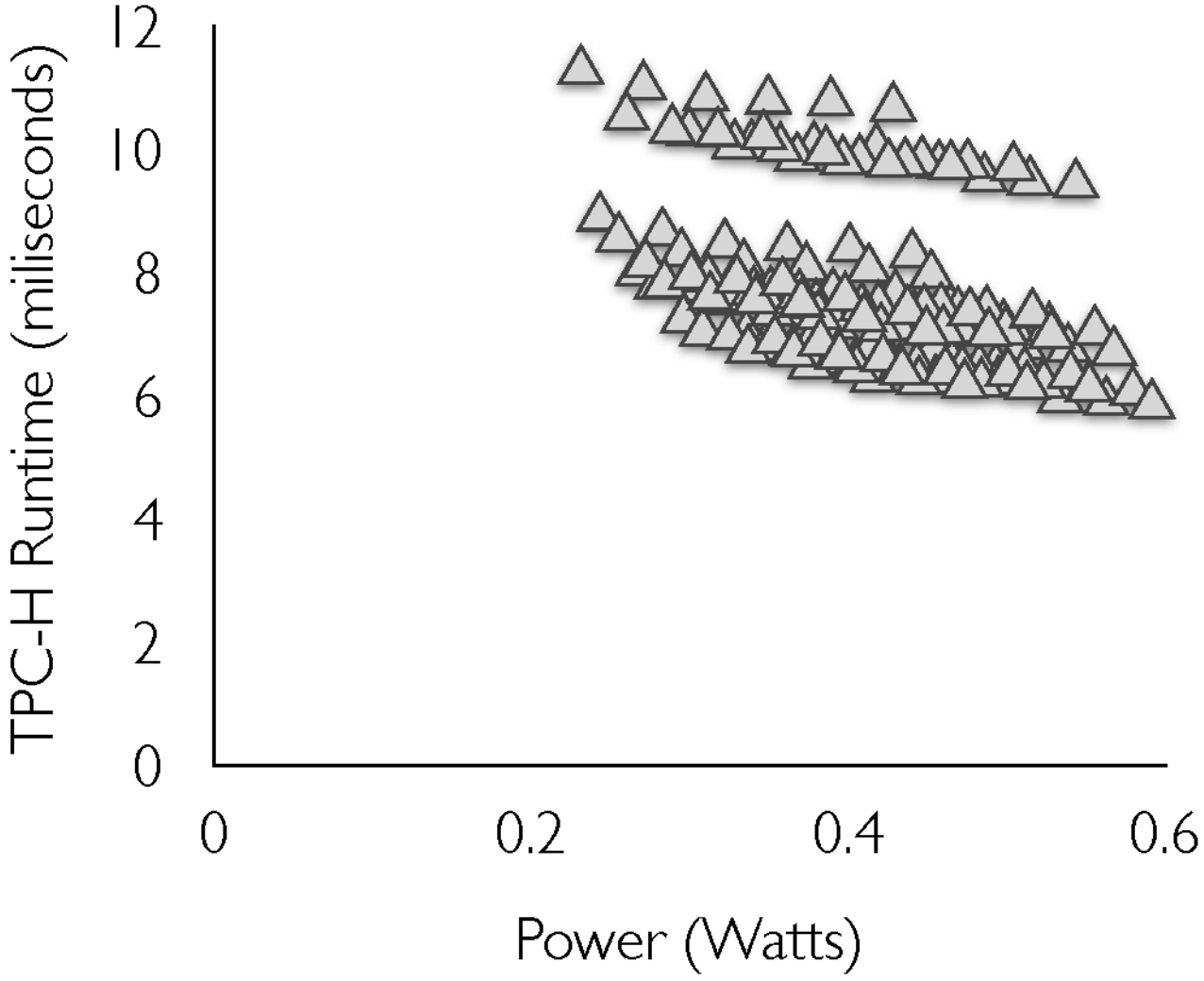
# Bounded Design Space



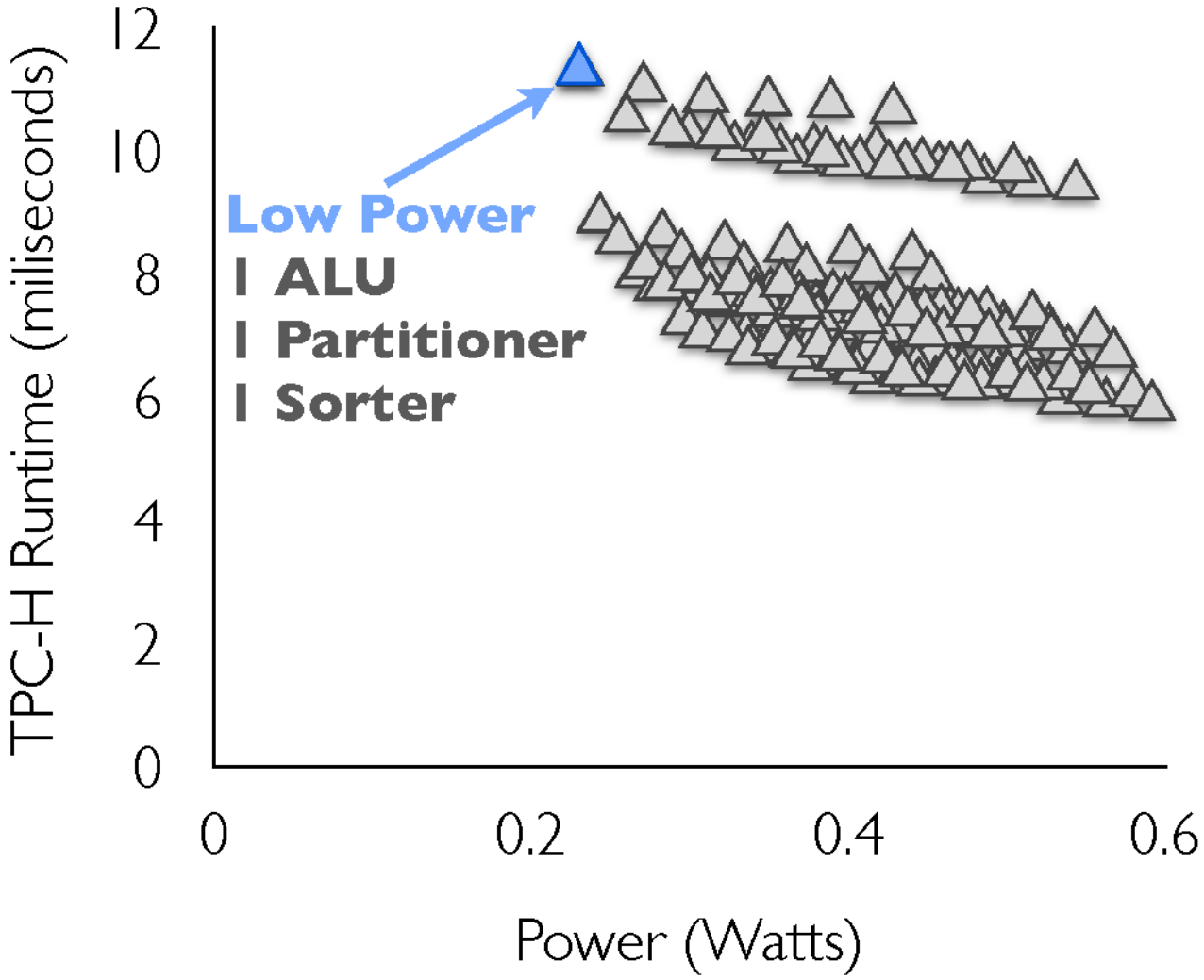
# Bounded Design Space



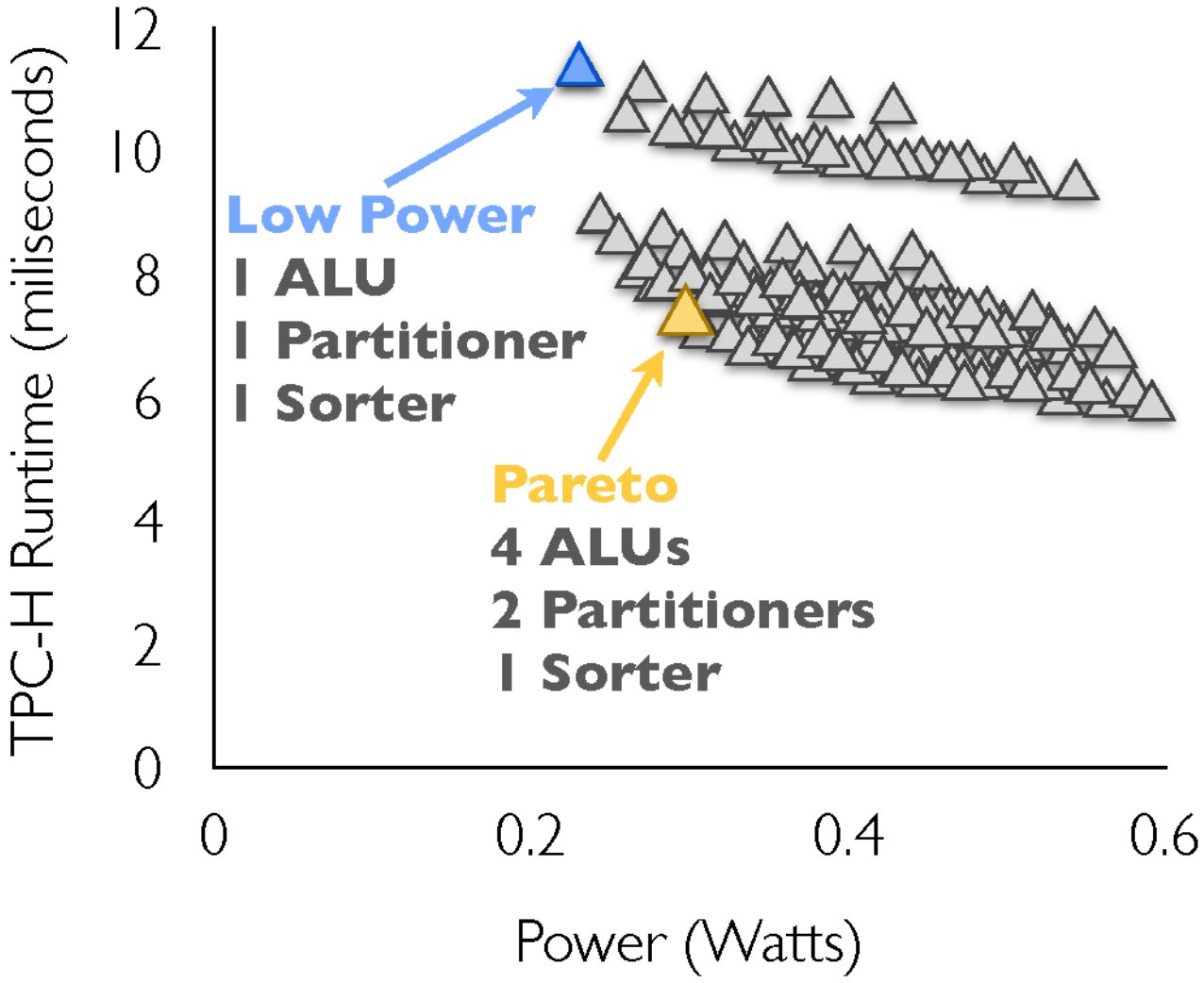
# Q100 Designs for Further Evaluation



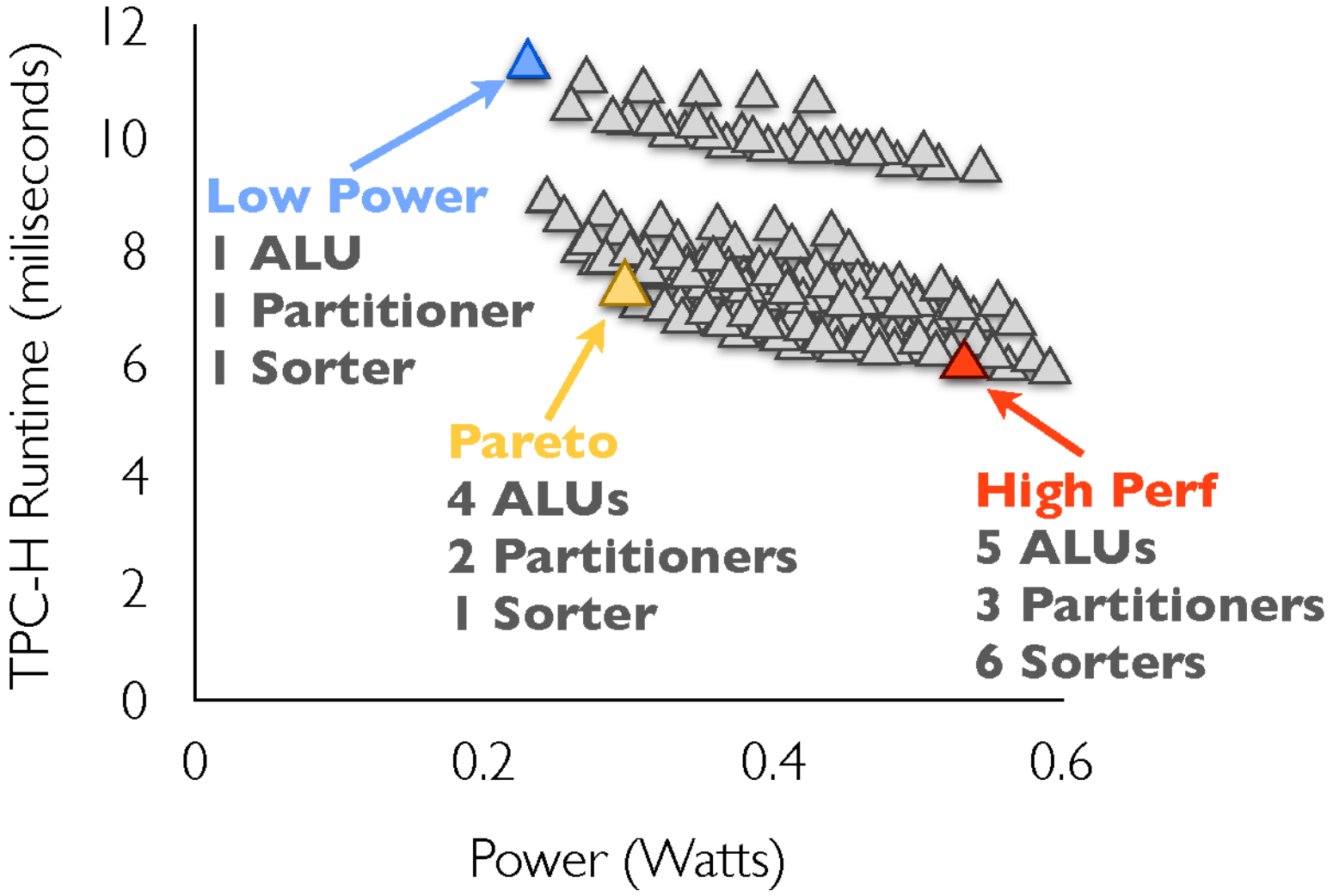
# Q100 Designs for Further Evaluation

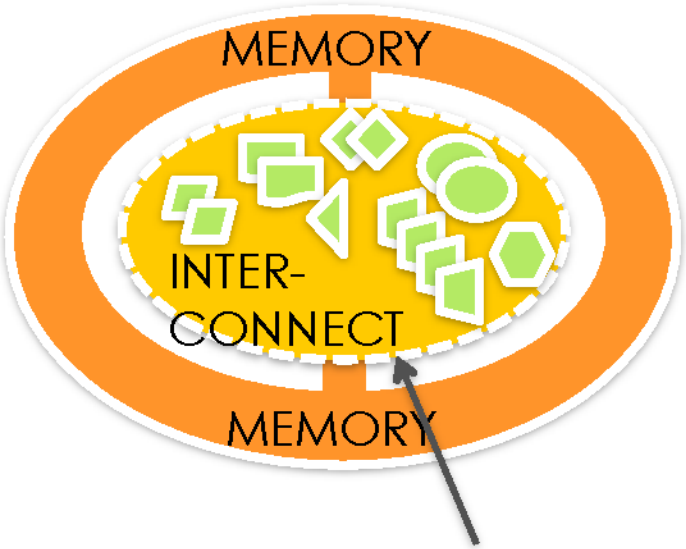


# Q100 Designs for Further Evaluation



# Q100 Designs for Further Evaluation



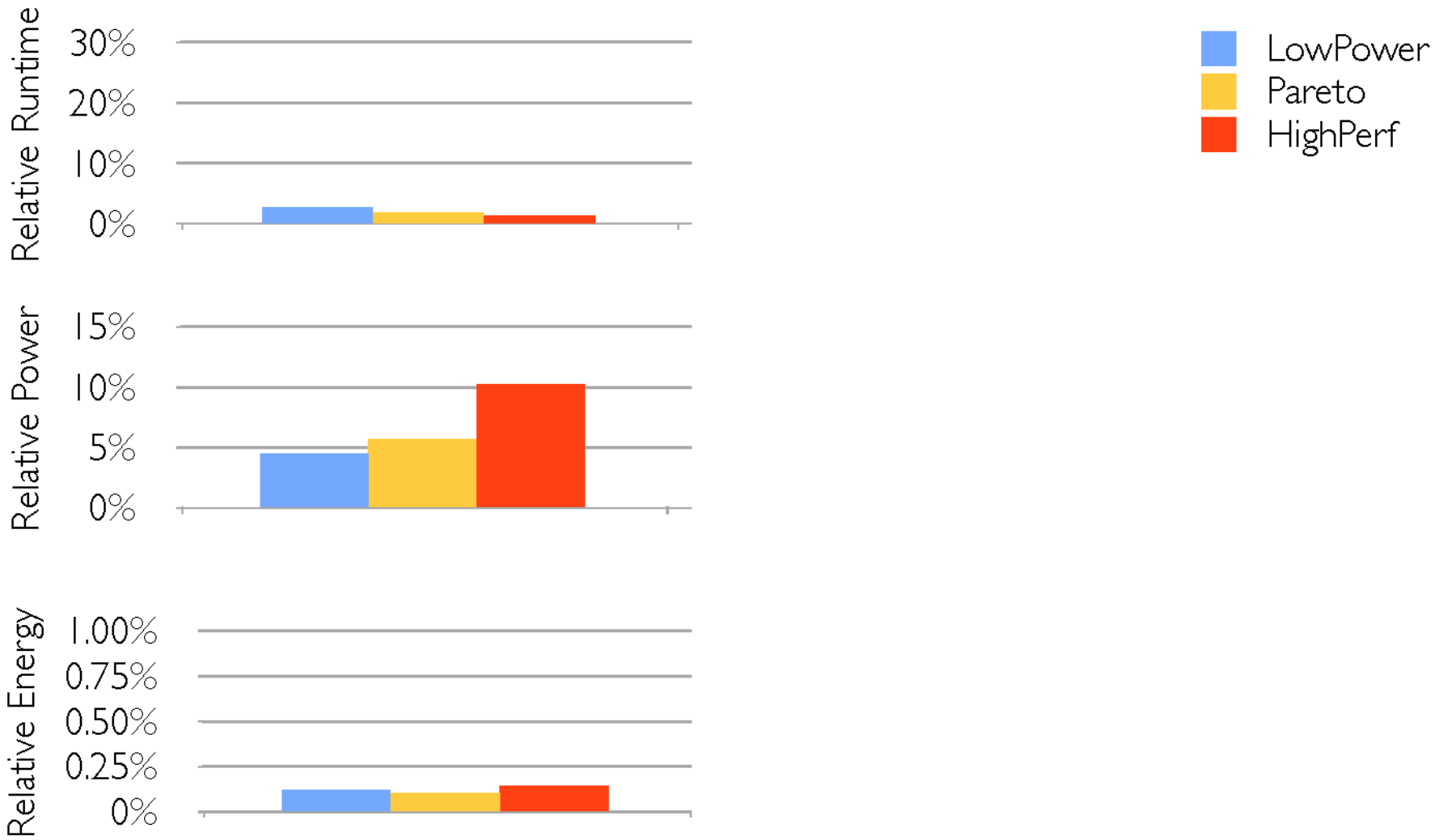



Is the Q100 performance and energy efficient?

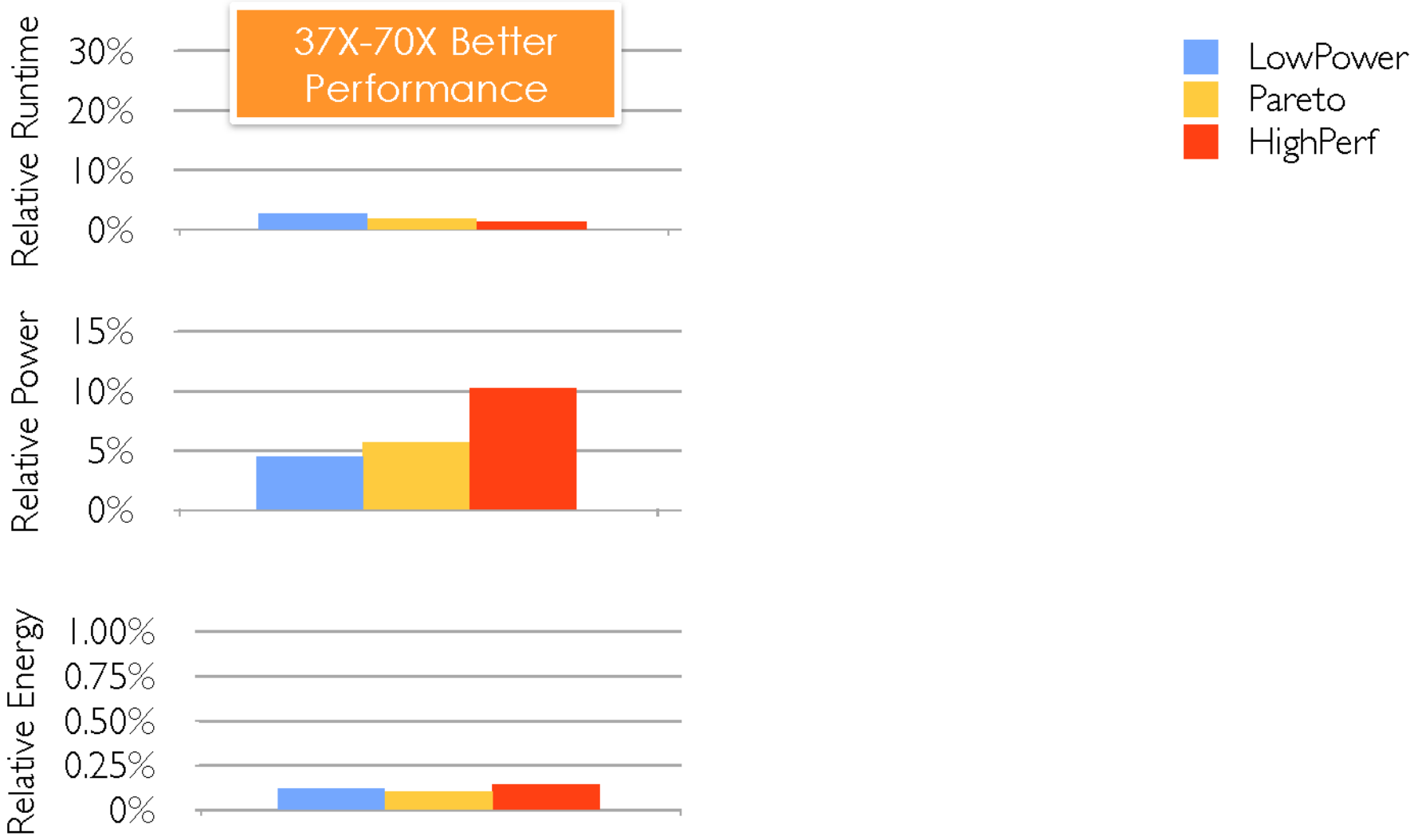
# Software Comparison Methodology

- MonetDB on Sandybridge server
- Energy Measurements:
  - Intel's Running Average Power Limit (RAPL) energy meters
  - Core domain only
  - Sample energy counters at 10ms intervals
  - Exclude machine idle power

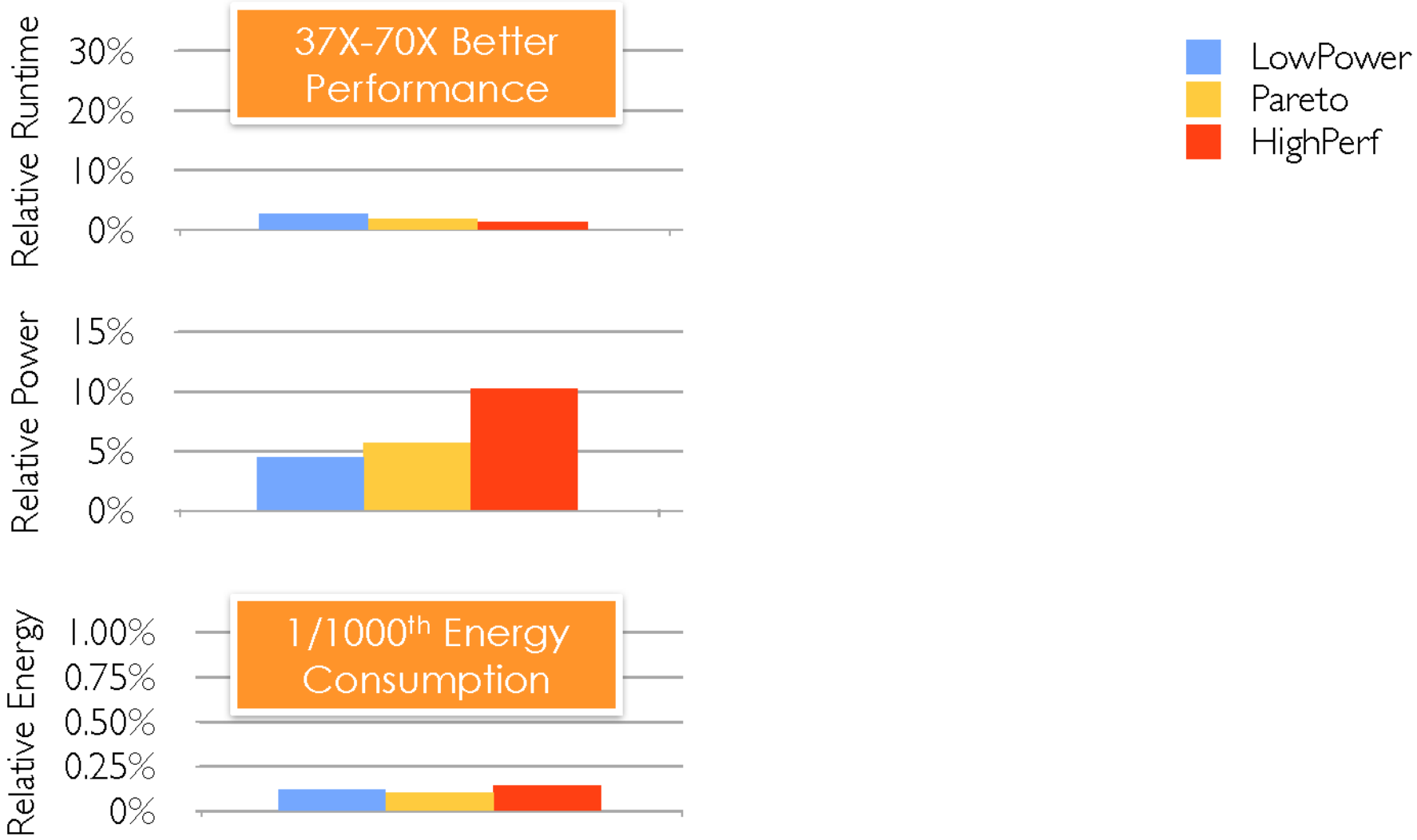
# Comparison with Software (MonetDB)



# Comparison with Software (MonetDB)

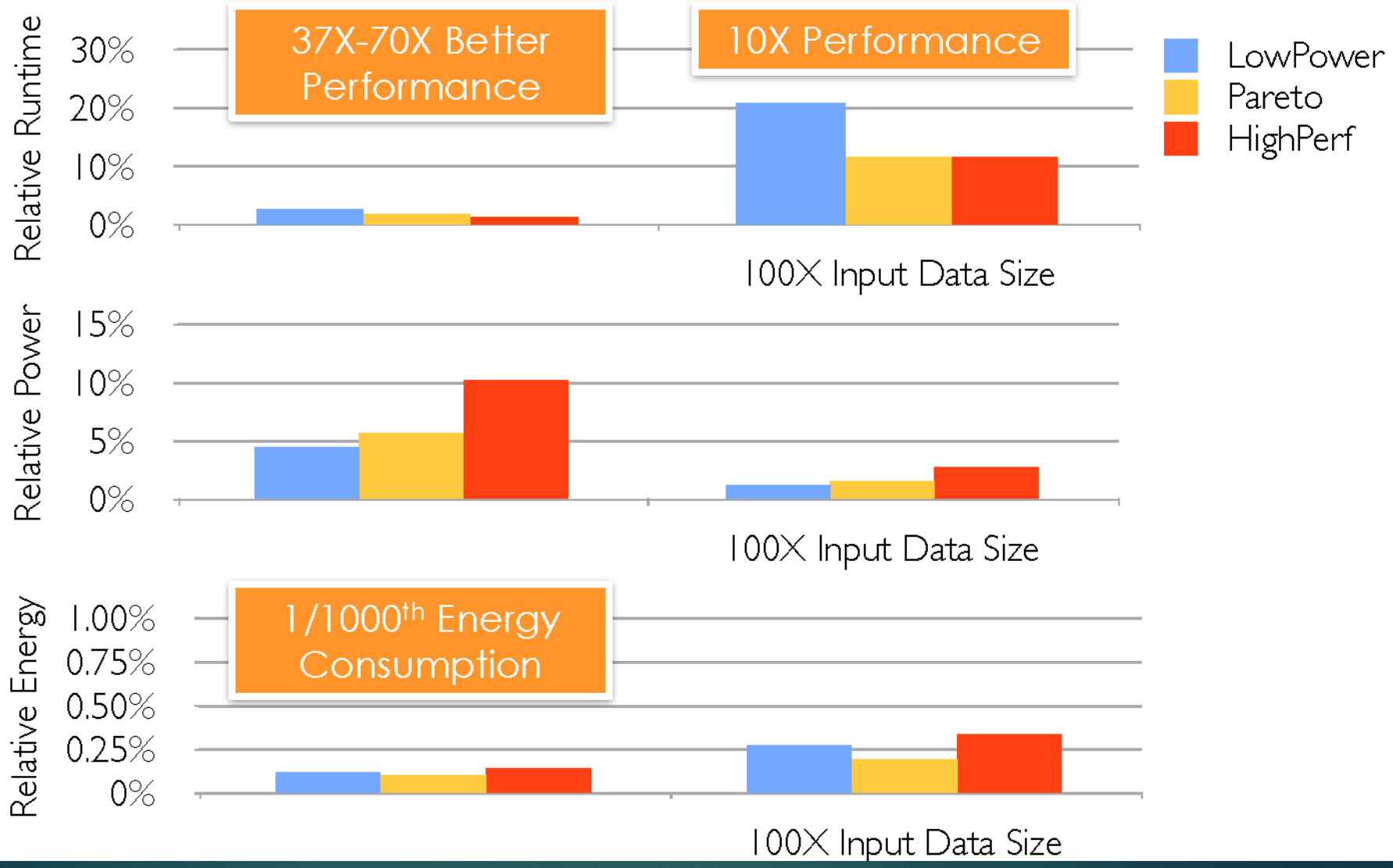


# Comparison with Software (MonetDB)

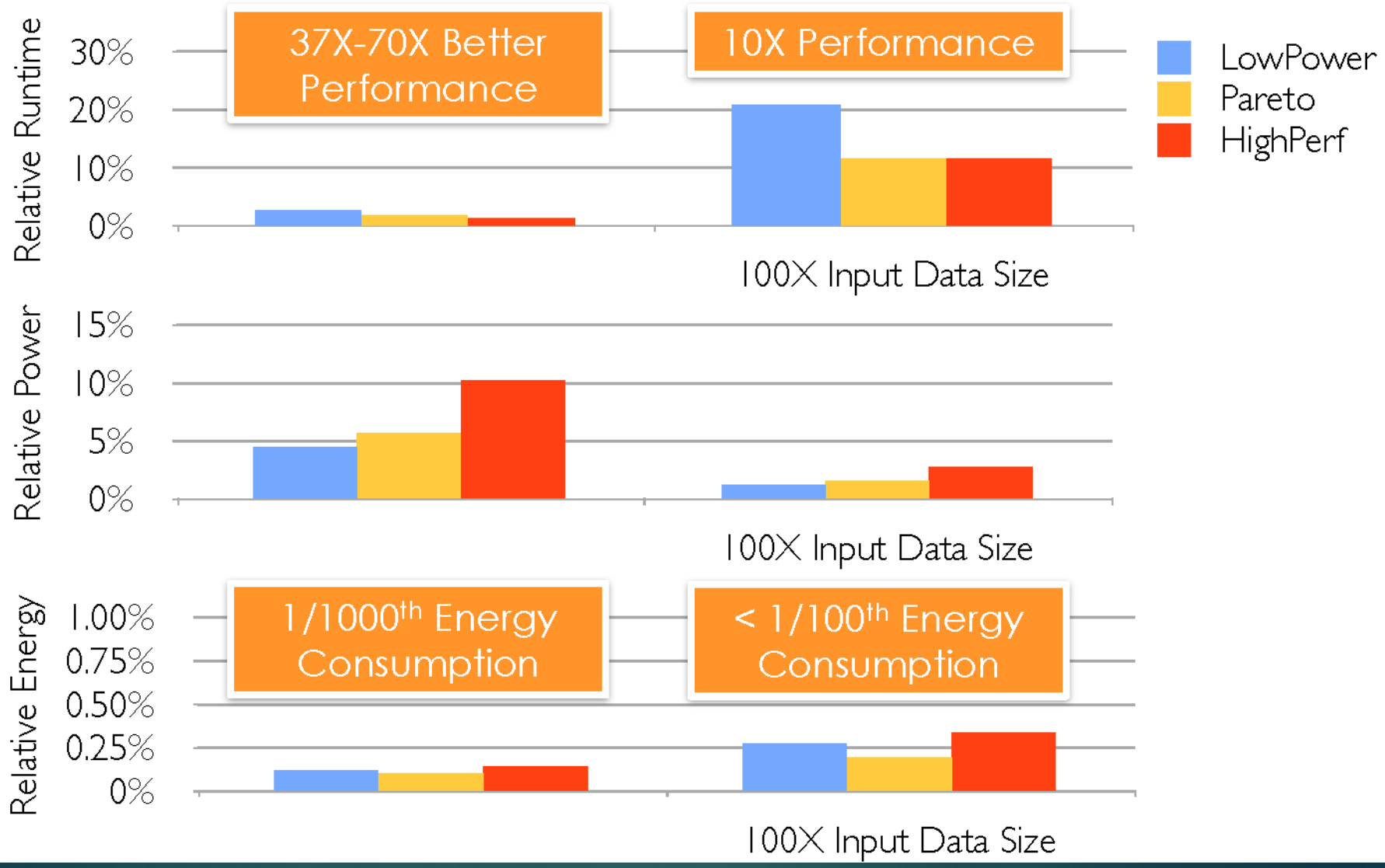


# Comparison with Software (MonetDB)

# Comparison with Software (MonetDB)



# Comparison with Software (MonetDB)



# Conclusions

- Q100 is a highly efficient domain-specific accelerator for analytical database workloads
- ISA exploits parallelism and streaming efficiencies
- At < 15% area and power of a Xeon core, a Q100 device gets exceptional performance and energy efficiency
- Exciting research opportunities for DPU

# Paper's implication

- Q100 specializes in big streaming data for analytics
- The first domain-specific circuit (vs query specific)
- Shows dramatic improvement in both energy (X1000) and performance (up to X70).
- But…

# Software Comparison Methodology

- MonetDB on Sandybridge server
- Energy Measurements:
  - Intel's Running Average Power Limit (RAPL) energy meters
  - Core domain only
  - Sample energy counters at 10ms intervals
  - Exclude machine idle power

# Datacenter Power

## Hardware subsystem power

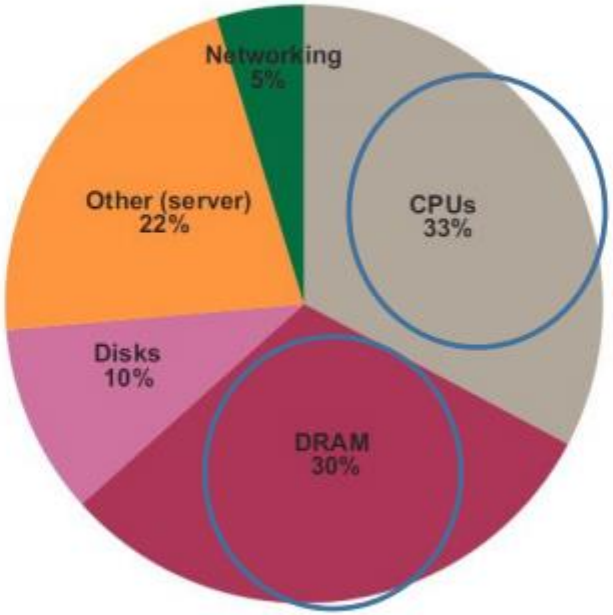
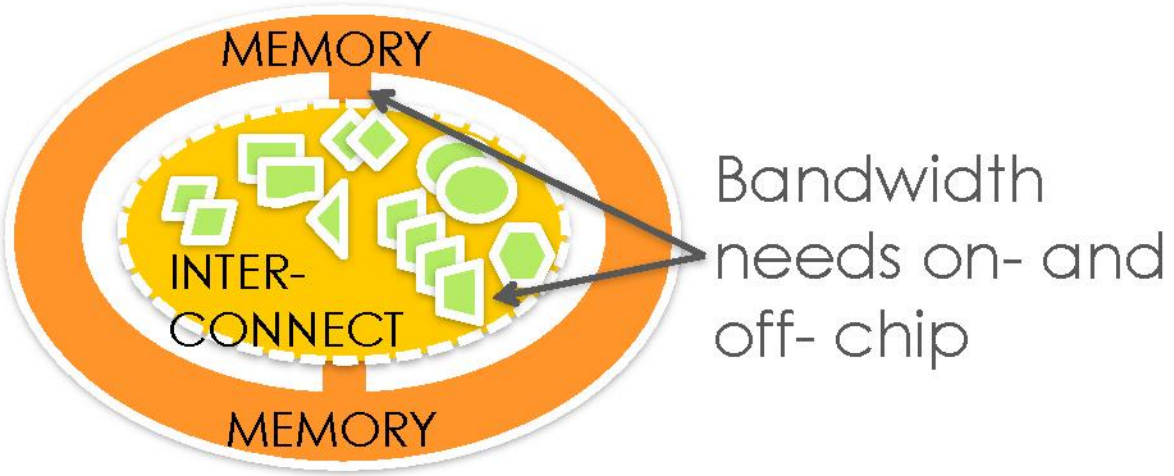


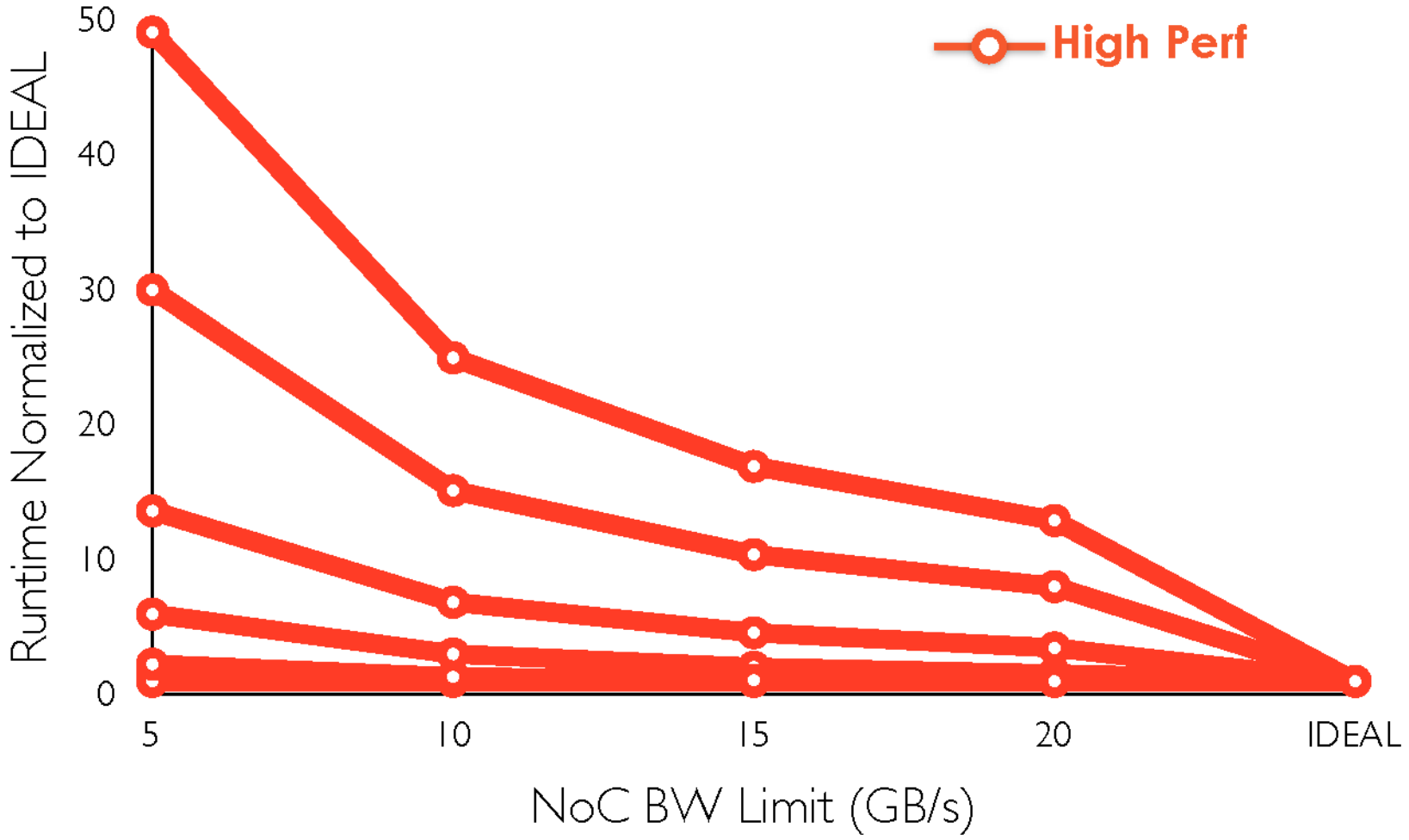

FIGURE 1.4: Approximate distribution of peak power usage by hardware subsystem in one of Google's datacenters (circa 2007).

# THE END

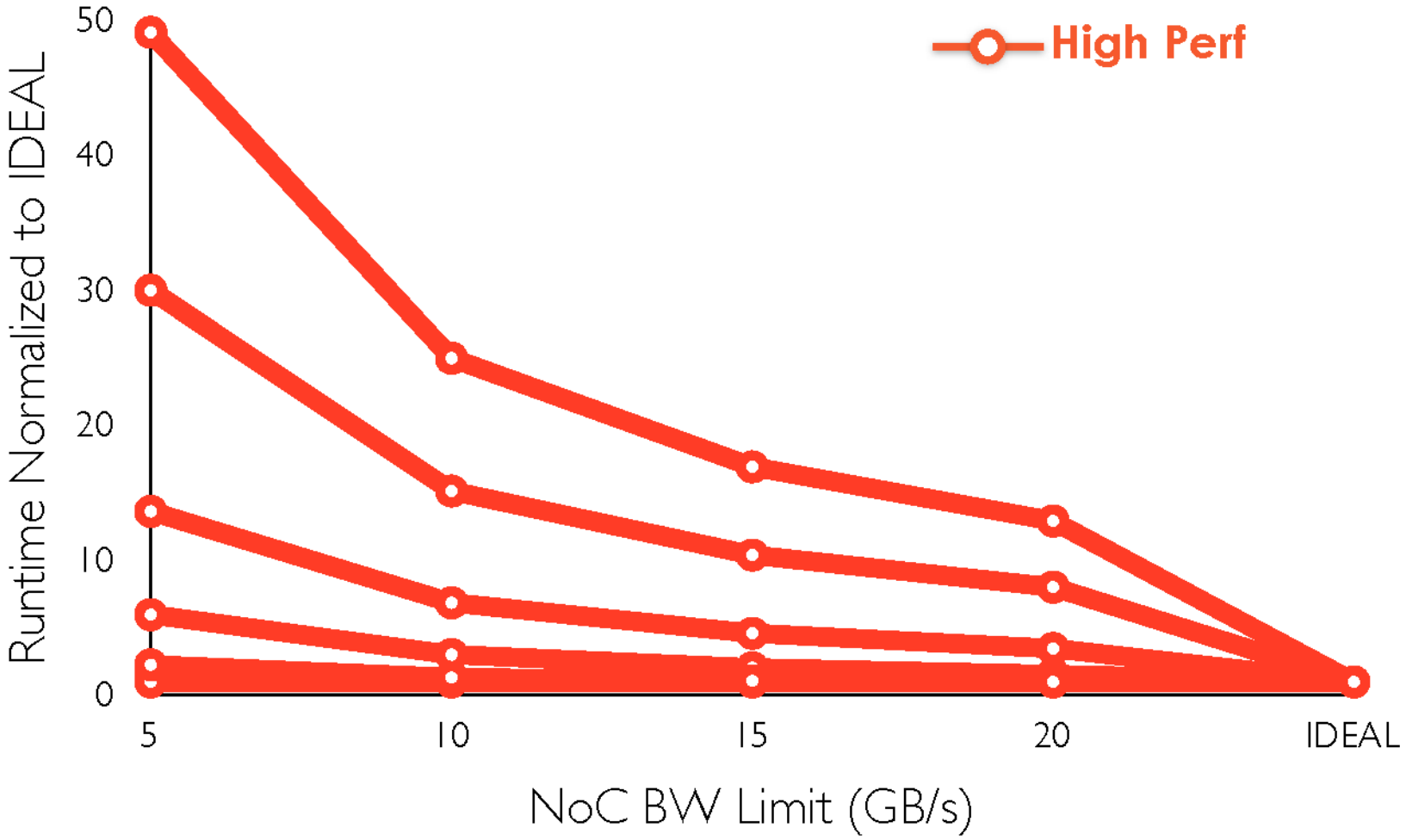
# Backup


MEMORY
INTER-
CONNECT
MEMORY
Bandwidth
needs on- and
off- chip

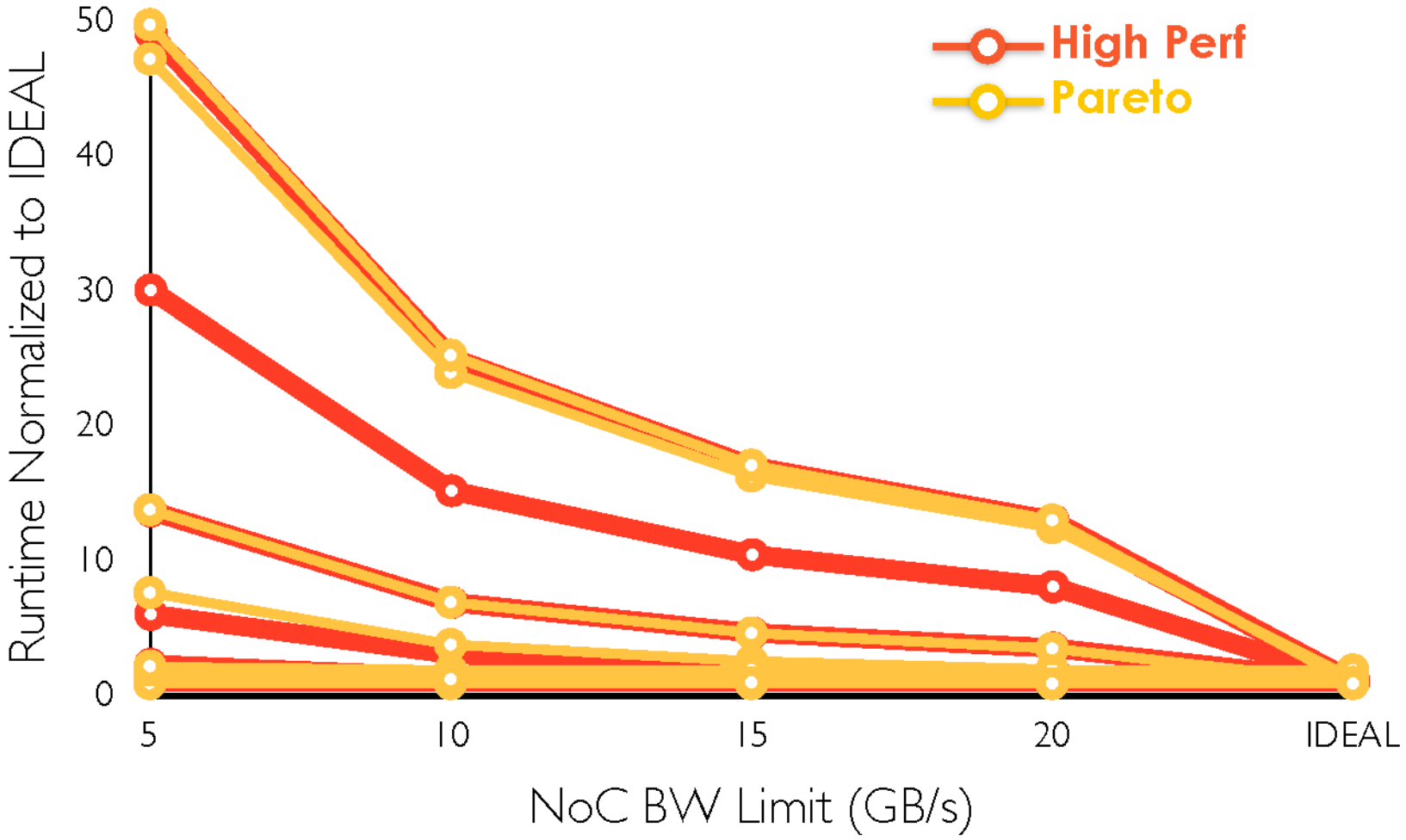
# Interconnect (Network on Chip) Bandwidth Needs



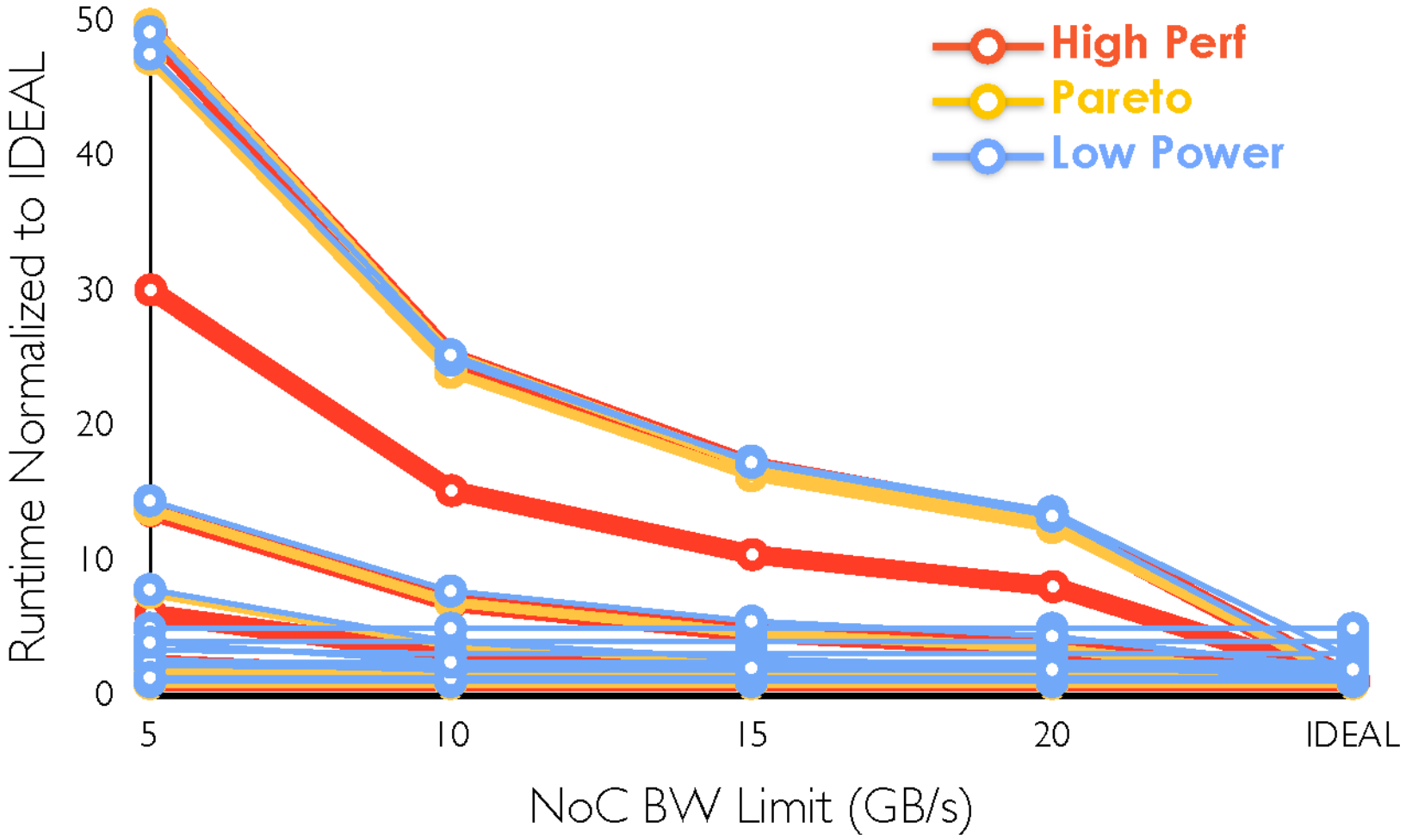
# Interconnect (Network on Chip) Bandwidth Needs



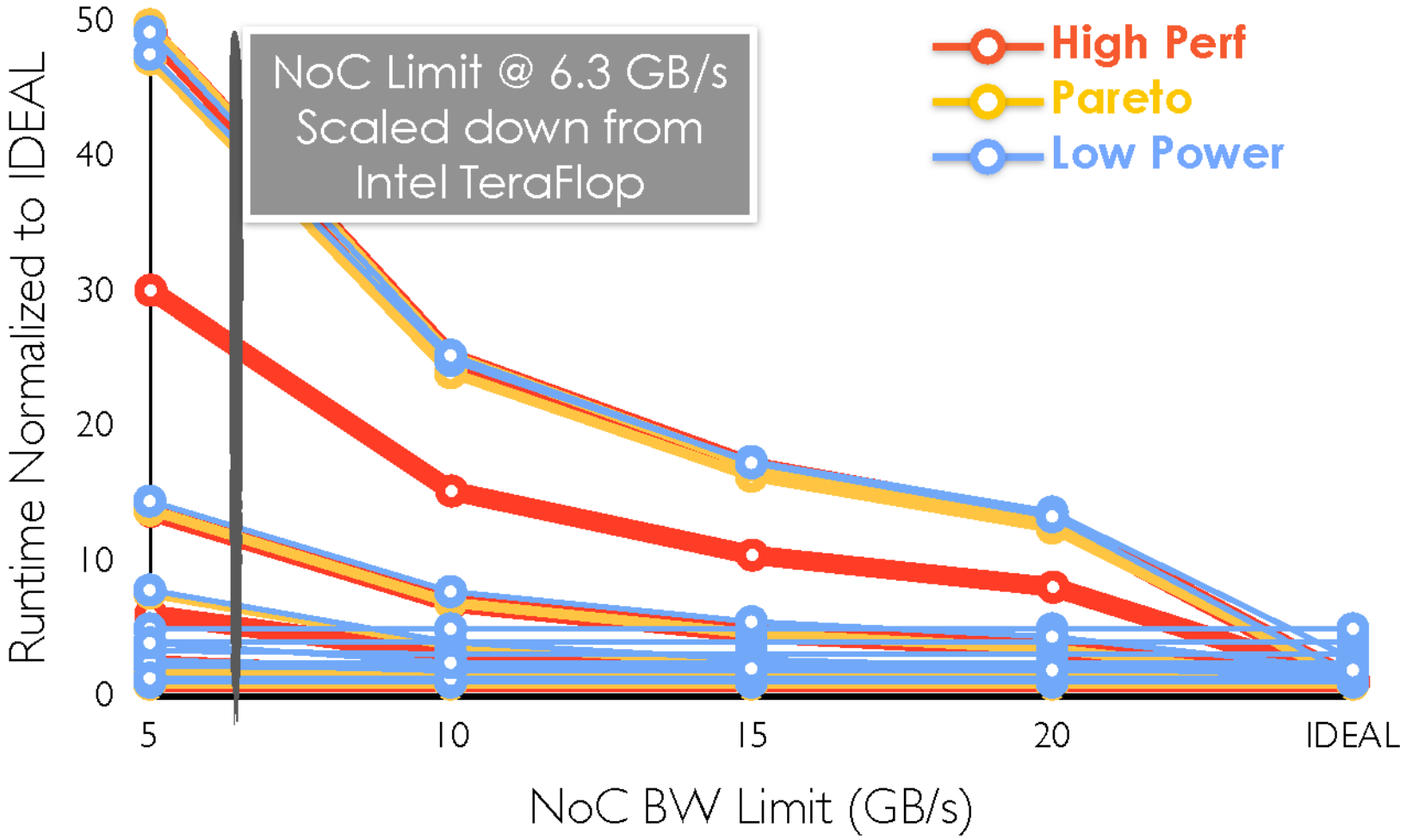
# Interconnect (Network on Chip) Bandwidth Needs



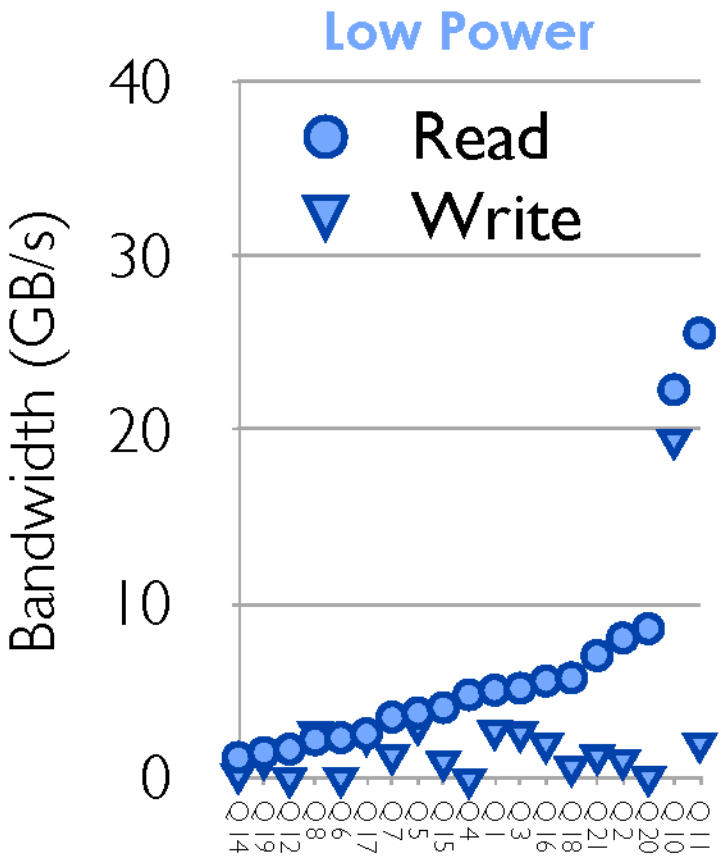
# Interconnect (Network on Chip) Bandwidth Needs



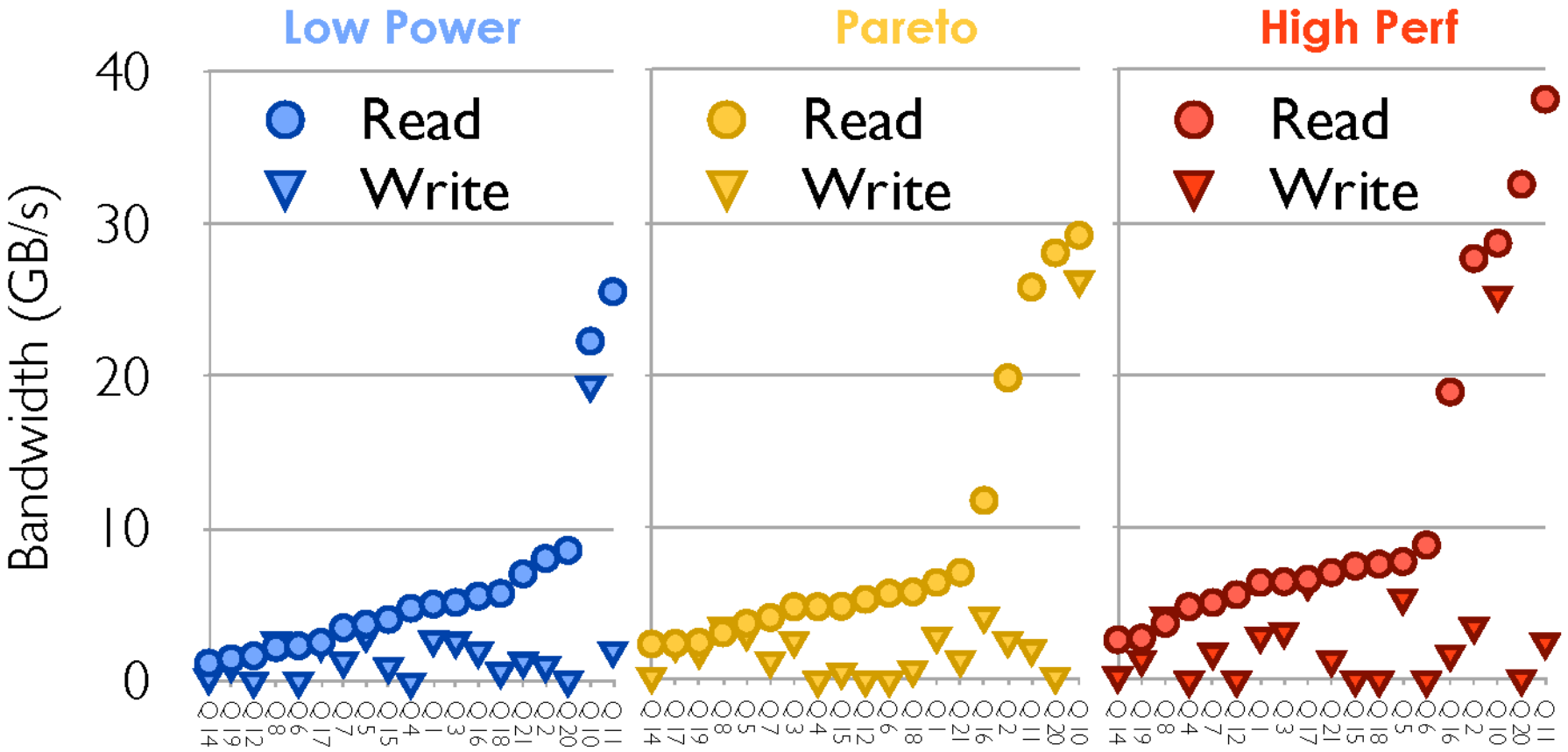
# Interconnect (Network on Chip) Bandwidth Needs



# Bandwidth to/from Memory

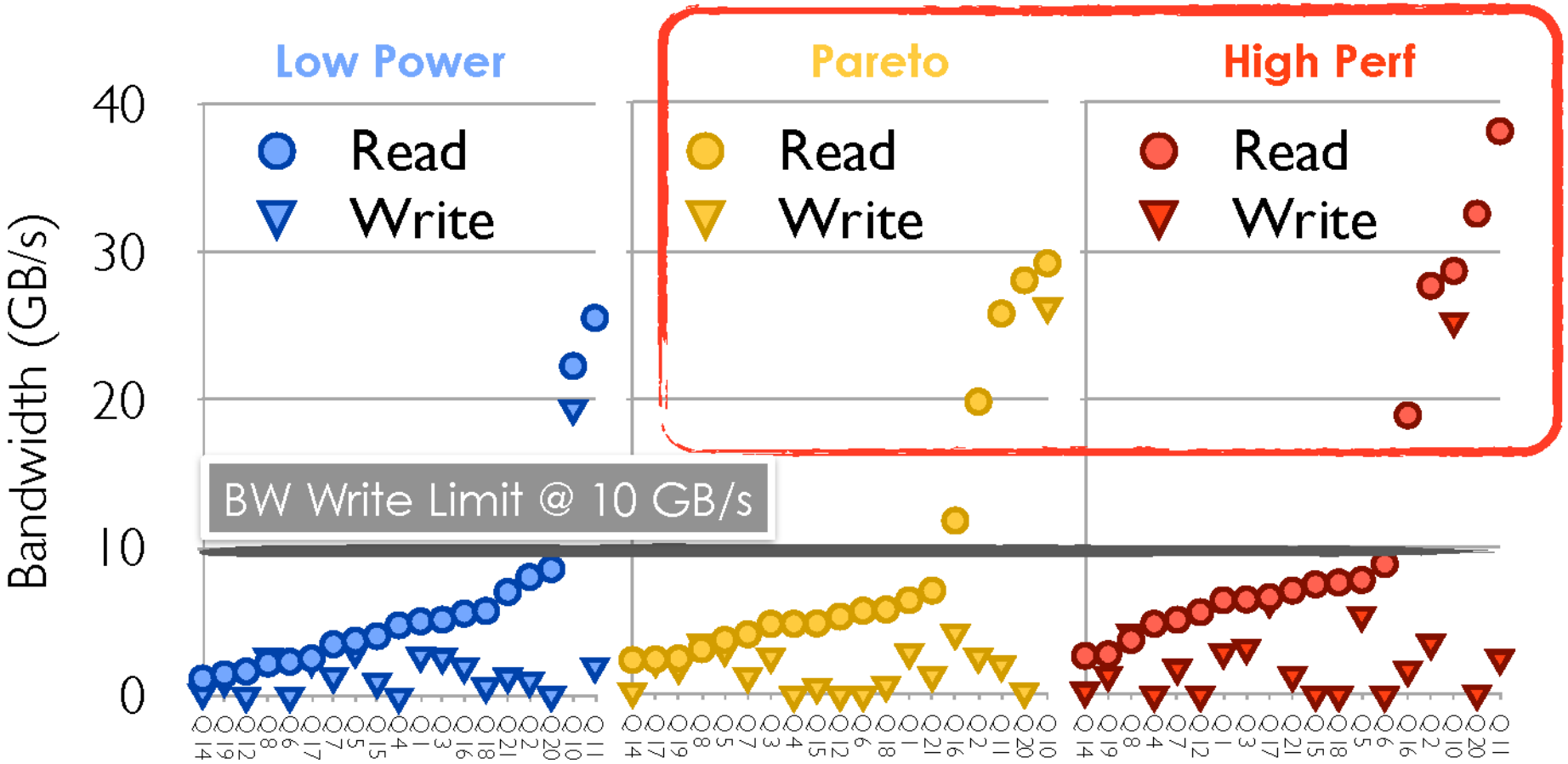


# Bandwidth to/from Memory



# Bandwidth to/from Memory

# Bandwidth to/from Memory



# Bandwidth to/from Memory