XDF XILINX DEVELOPER FORUM

Deepgreen MPP with FPGA: A supercharged Greenplum Data Warehouse solution

Presented By

VITESSE DATA

Feng Tian
Founder

XILINX

Agenda: Building Next Gen Data Analytics Platform

- Why NOW?
- How to Accelerate Data Warehouse with FPGA
- Use Cases

It’s Time for a complete rewrite

The End of an Architectural Era
(It’s time for a complete rewrite)

by

Michael Stonebraker

ACM Turing Award Winner

- **New Application Landscape**
- **Rich Data**
  - Text
  - IoT, Geospatial
  - Media
- **Intelligent Data**
  - Query getting more complex
  - Geospatial
  - Machine learning/Data mining
  - AI/Deep learning

Time for a complete rewrite: Hardware Trend

- **Storage Hierarchy**
  - Big Memory
  - SSD
  - → Plenty of Bandwidth
- **Network**
  - 10, 100 GigE is common
  - → Plenty of Bandwidth

>>>> Today, most Data Workload is bottlenecked on CPU <<<<

- **FPGA can relief CPU**

A New Golden Age for Computer Architecture

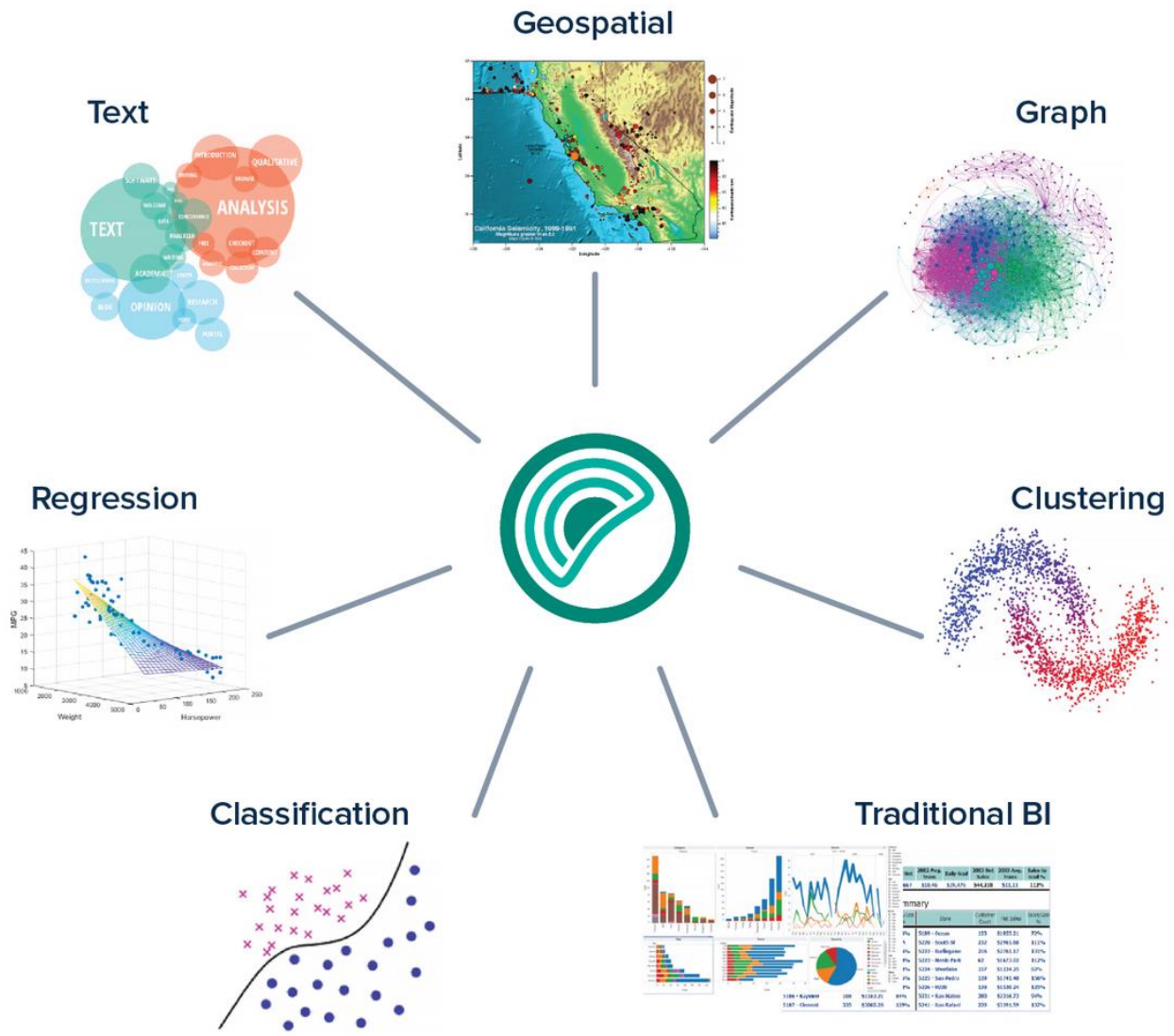
AWARDS & RECOGNITION

John Hennessy and David Patterson Receive 2017 ACM A.M. Turing Award

- **Domain Specific Hardware/Software Co-Design**
- **Enhanced Security**
- **Open Instruction Set**
- **Agile Chip Development**

Data, Data Everywhere …

Actionable Insight



Deepgreen DB: a better Greenplum Data Warehouse

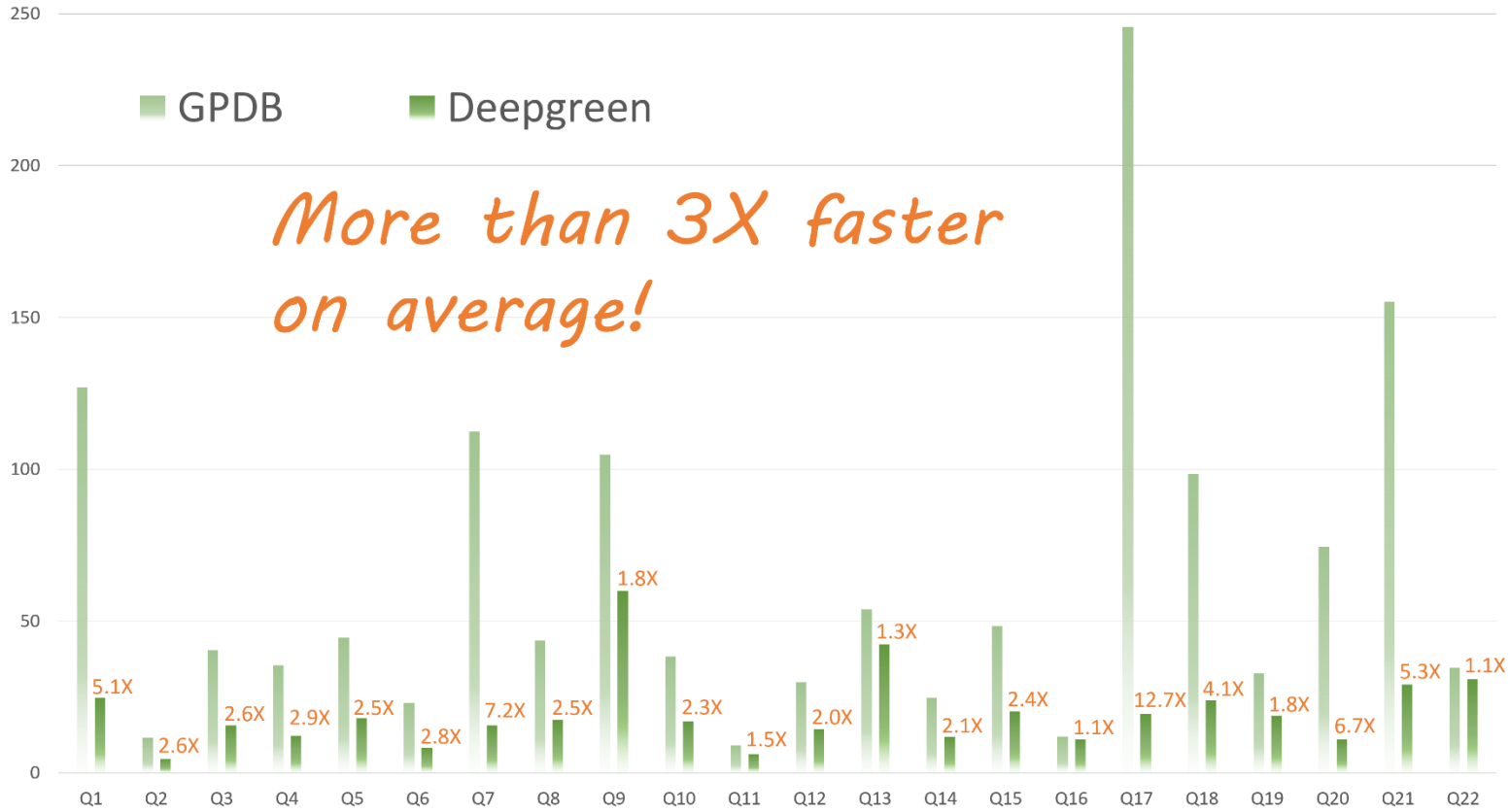
- **Greenplum**
  - Field tested with widespread adoption in Telco, Financial, Government, Retails, …
- **Deepgreen**
  - We squeezed every bit of juice out of x86 CPUs
  - 100% compatible
  - Zero code-change to switch
  - Complete rewrite of Query Execution Engine
    - LLVM JIT
    - SIMD
    - Switch binary (without reloading data) to get 3-5x performance boost.
    - *Now even faster with FPGA!*

Deepgreen: FPGA Hardware Acceleration

- **LLVM JIT + SIMD**
  - We squeezed CPU dry

- ***Next frontier:***

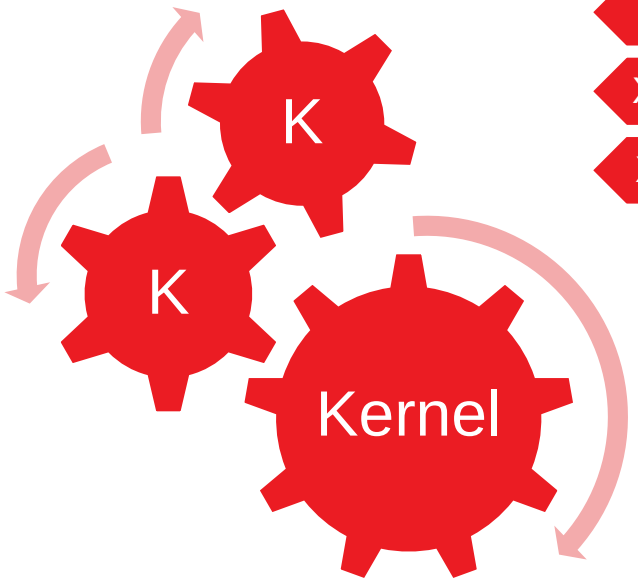
  >> FPGA



FPGA In Deepgreen

Challenges	Our Approach
➢ Memory is big, but not big enough	➢ Identify the bottleneck
➢ Throughput vs Latency	➢ New algorithm tuned for FPGA
➢ Multi-CPU/Core	➢ Offload to FPGA, none preemptive
➢ Multiuser environment	➢ XLIW: eXtra Long Instruction Word

XLIW: eXtra Long Instruction Word

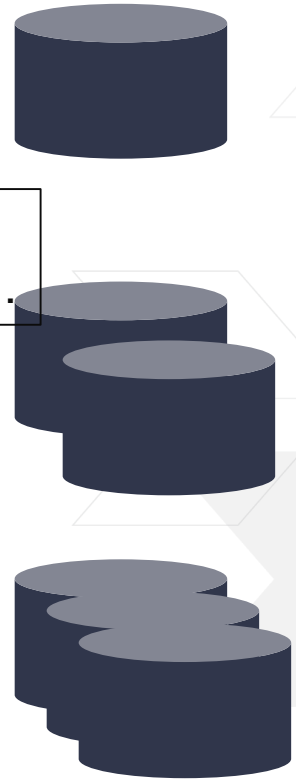


XLIW XLIW XLIW
XLIW XLIW XLIW
XLIW XLIW XLIW

Reslut R R
Result R R
Result R R

XLIW: Hasher
Data, Data, Data …

XLIW: Hasher
Data, Data, Data …

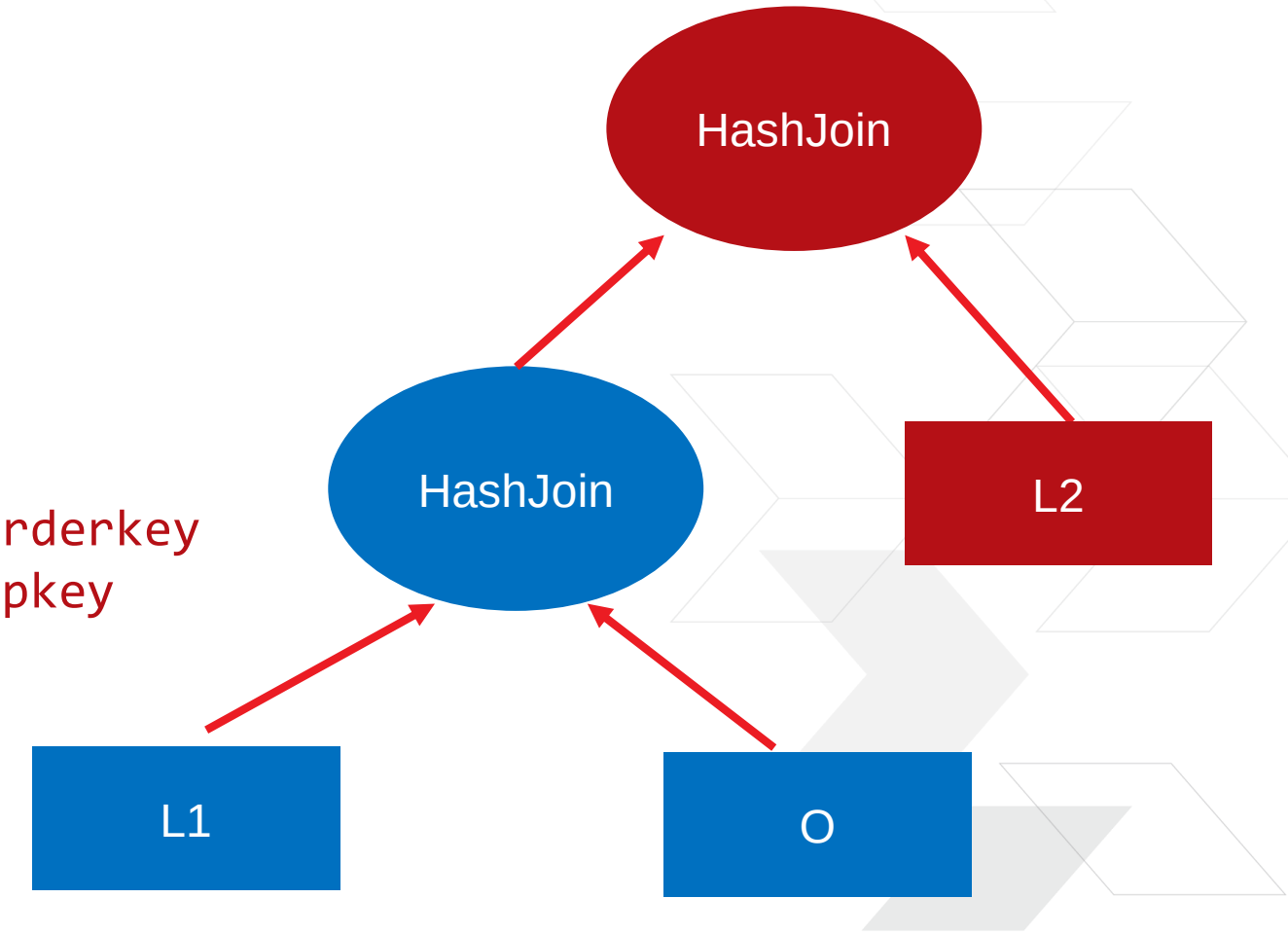


Use Case 1: Hash Join

```
SELECT count(*)
FROM lineitem L1, orders O
WHERE O.o_orderkey = L1.l_orderkey

AND EXISTS (

        SELECT *
        FROM lineitem L2
        WHERE L2.l_orderkey = L1.l_orderkey
        AND L2.l_suppkey <> L1.l_suppkey
)
```



Use Case 1: Hash Join Implementation

Hash Join

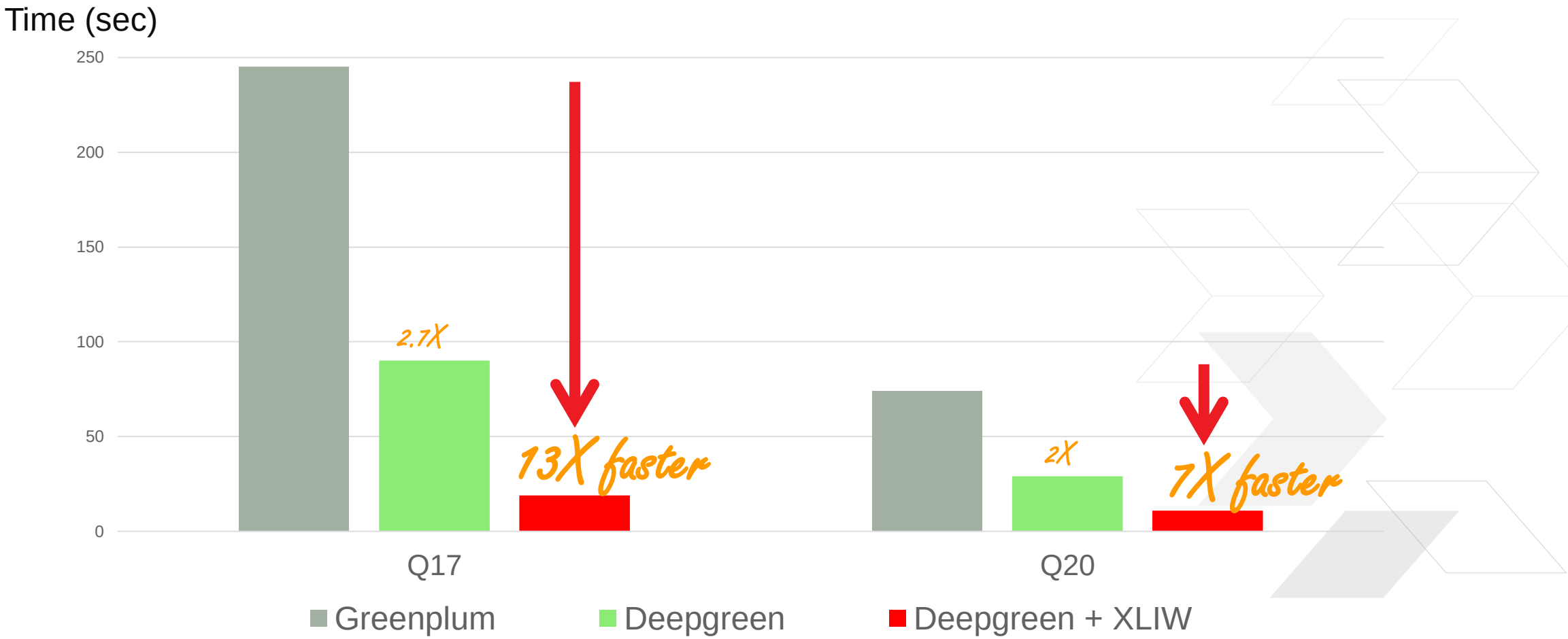
- HJ Algorithm (expressed trivially):
  1. Scan left side and build hash table
  2. Scan right side, and probe hash table
  3. Output all hits
- Lots of records joined
- Hash table is not cache friendly

XLIW for Hash Join

- Pack a lot of records of left side, send to FPGA to compute hashes
- Instead of using hash table, we sort the hashes using a very fast radix sort. (10x faster than quicksort)
- Pack a lot of records from right side, send to FPGA to compute hashes. Sort with radix sort
- Merge
- It is a hybrid hash/sort merge join

Use Case 1: Hash Join Performance

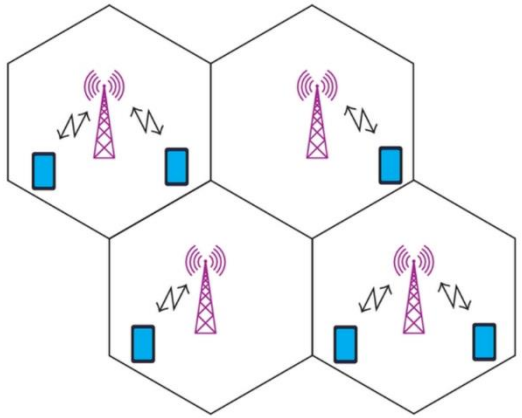
TPCH Q17 and Q20 on AWS F1

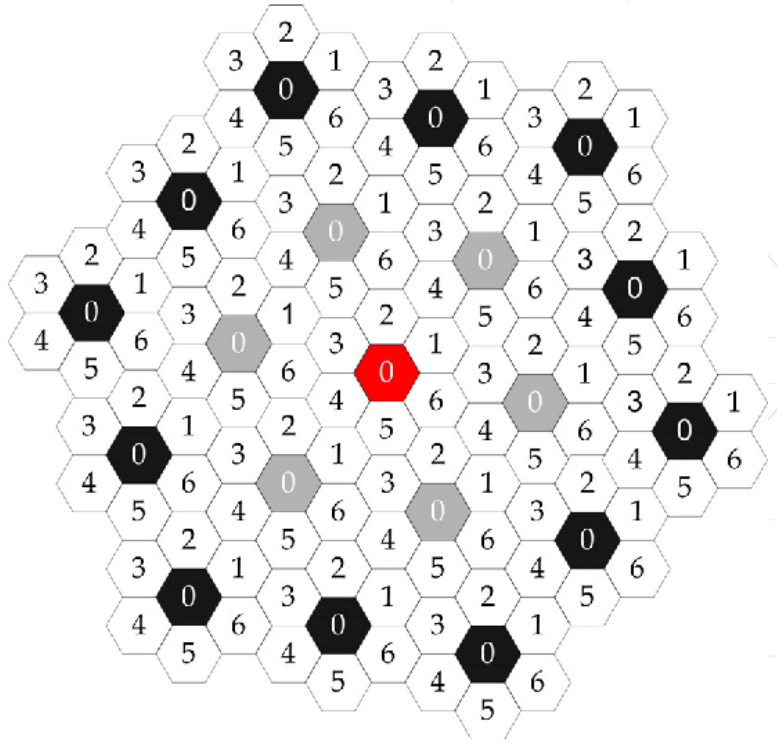


Use Case 2: GeoSpatial Join

```
/* count devices covered by each cell tower */

SELECT t.tower_id, count(*)
FROM towers t, devices d
WHERE ST_intersects(t.area, d.location)
GROUP BY t.tower_id
```





Use Case 2: GeoSpatial Join

Greenplum + PostGIS

- PostGIS is the GeoSpatial extension of PostgreSQL/Greenplum/Deepgreen
- Naïve Join will never finish

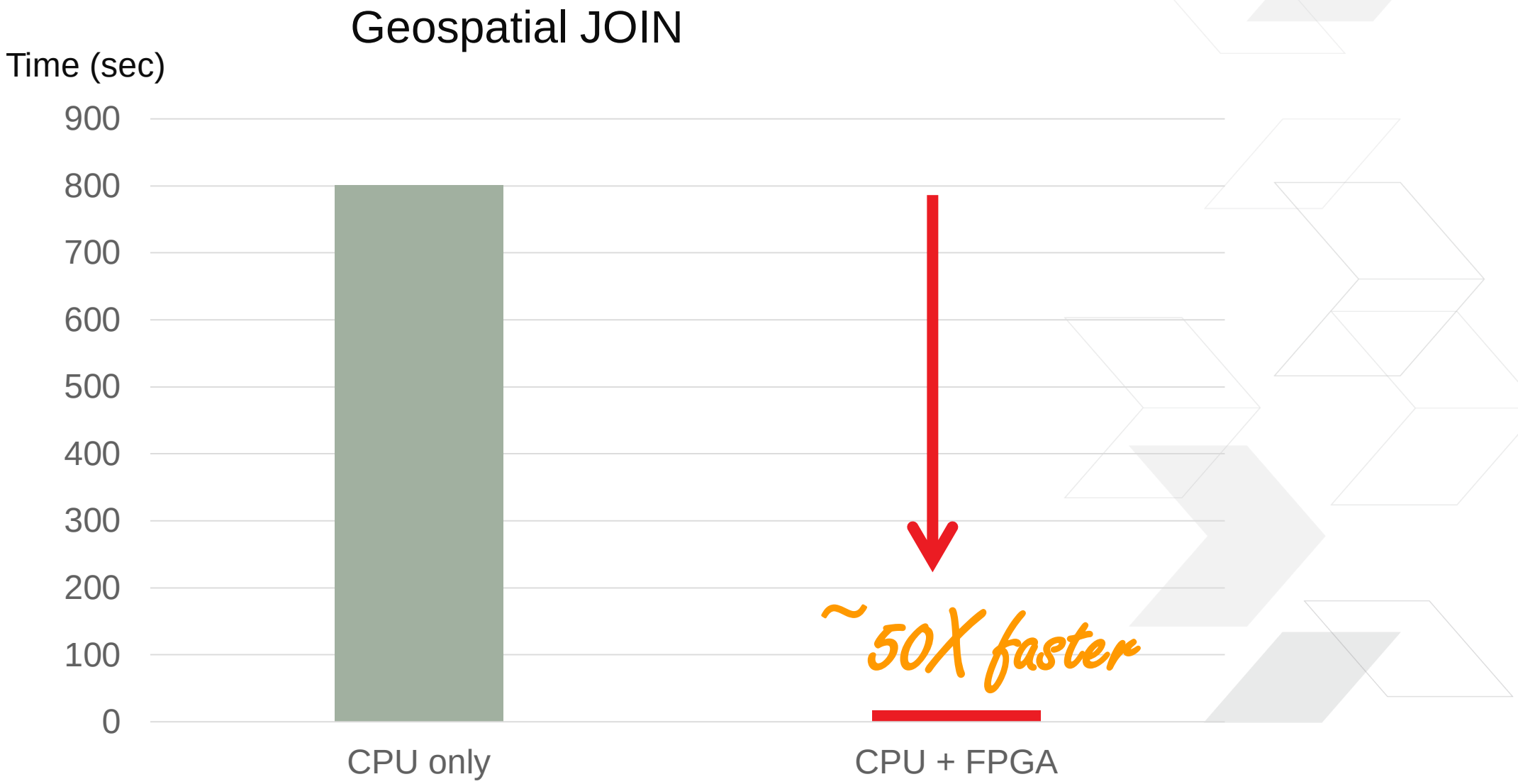
- Build index (R-tree)
- Index Nestloop Join
  - For each polygon, using index to lookup points nearby
  - Check the intersects condition

GeoSpatial Join + XLIW

- Do not use index

- Scan outer loop, build an in-memory data structure
  - Still expensive operation, but cheaper than compute intersection (like building an R-tree)
- Scan inner loop, probing the in memory data structure (like probing R-tree)
- Check intersection
  - This step is dominating execution time
  - Build/Pack XLIW instruction, send to FPGA

Use Case 2: Performance



Use Case 3: Adding Intelligence (Available Soon)

- **An XLIW for data mining/machine learning**
- **Deepgreen Transducer Framework**
  - Allow user to embed C/Java/Go/Python code in SQL
  - Interleaved with SQL Engine code
  - First class citizen, optimized by query optimizer, executed in parallel, streaming data to/from SQL query operators like Sort/Join/Aggregate
- **ML libraries, Tensor Flow**
  - For example, Deep Neural Network in FPGA

Current Status and Future Directions

- **Deepgreen DB on AWS F1**
  - See our demo
  - On AWS Market Place (2018)
- **On-prem deployment with Alveo Accelerator Card**
  - Looking for early customers
- **We are just scratching the surface**
  - More use cases, endless opportunities
  - More to squeeze

Thank you &
Stay tuned!

Adaptable.
Intelligent.

VITESSE DATA

XDF XILINX DEVELOPER FORUM

XILINX