GPU ACCELERATED DATABASE MANAGEMENT SYSTEMS

PRESENTED BY
HARINATH AMASA
CSU ID: 2697292
What we will talk about..

• Current problems
• GPU
• What are GPU Databases
• GPU Programming model
• Demo
• GPU performance pitfalls
• The future
Evolution of Data Processing

1990 - 2000s: DATA WAREHOUSE
RDBMS and Data Warehouse technologies enable organizations to store and analyze growing volumes of data on high performance machines, but at high cost.

2005...: DISTRIBUTED STORAGE
Hadoop and MapReduce enable distributed storage and processing across multiple machines. Storing massive volumes of data becomes more affordable, but performance is slow.

2010...: AFFORDABLE IN-MEMORY
Affordable memory allows for faster data read and write. HBase, HANA, and MemSQL provide faster analytics. At scale compute processing now becomes the bottleneck.

2016...: GPU-ACCELERATED COMPUTE
GPU cores bulk process tasks in parallel—far more efficient for compute-intensive tasks than CPUs.
LET'S TALK BIG DATA
Data is STILL growing exponentially

CPU technology - linear progress

Up to 1-4TB

Up to 10TB

Hundreds of TB
(Sometimes even petabytes of data)
coming in at a rate of multiple terabytes per day
WE’RE IN THE PETABYTE AGE

• Petabyte datasets are now the norm
• Even small companies have dozens of terabytes of data for analysis

• Some outliers have more:
  – CERN processes 1 petabyte per day, stores 530 PB total
  – In 2012, Facebook analyzed 5 petabytes per day, stores estimated a few exabytes
  – The NSA might hold 12 exabytes
ARE WE ONLY ANALYZING THE TIP OF THE ICEBERG?
Base Configuration
Scale Up: Add More CPUs
NVIDIA - INVENTOR OF THE GPU

- NVIDIA Invented the GPU in 1999, with over 1 Billion shipped to date.
- Initially a dedicated a graphics processor for PCs, the GPU’s computational power and energy efficiency led to it quickly being adopted for professional high performance computing and enterprise data analytics.
- In 2007, NVIDIA launched the CUDA® programming platform, and opened up the general purpose parallel processing capabilities of the GPU.
Graphics Processing Unit (GPU)

• Specialized processor, can be programmed similar to CPUs
• GPUs achieve high performance through massive parallelism
  → Problem should be easy to parallelize to gain most from running on the GPU
• Single Instruction, Multiple Data (SIMD): Each multiprocessor only has a single instruction decoder
  → Scalar processors execute the same instruction at a time
• Optimized for computation
Example: Fermi Architecture of NVIDIA

CPU

Main Memory
~30GB DDR3

Host System

3x 10.57 GB/s

Device Memory
2GB GDDR5

121 GB/s

48 Scalar Processors
Memory Controller
Instruction Decoder

Graphics Card

7 Multiprocessors
On-Chip Shared Memory 64kB

1 TB/s

x16 PCIe Express 2.1 Bus

8 GB/s
Graphics Processing Unit

CPU Memory

CPU

GPU Memory

GPU

PCI Express bus
TESLA P100 – 3584 CUDA CORES
What are GPU Databases?

• A GPU database is a database, relational or non-relational, that uses a GPU to perform some database operations
• Most of the GPU databases tend to focus on analytics, and they’re offering it to a market that was oversold on Hadoop for Big Data analytics
• And they’re typically pretty fast

And they’re not only disrupting the in-memory crowd
• GPU databases are more flexible in processing many different types of data, or much larger amounts of data
Why GPUs in big data?

• High core count allows offloading of ‘heavy’ stuff like JOINs, ORDER BY, GROUP BY from the CPU to the GPU
• Compression and Decompression processes reduce PCI and disk I/O. These are basically free on the GPU
• Can also use GPU to do computationally intensive operations like deep learning, cryptography.
GPU Acceleration Overcomes Processing Bottlenecks

GPUs are designed around thousands of small, efficient cores that are well suited to performing repeated similar instructions in parallel. This makes them well-suited to the compute-intensive workloads required of large data sets.

- **4,000+ cores per device in many cases, versus 8 to 32 cores per typical CPU-based device.**

- **High performance computing trend to using GPUs to solve massive processing challenges.**

- **Parallel processing is ideal for scanning entire dataset & brute force compute.**

- **GPU acceleration brings high performance compute to commodity hardware.**
Benefits of GPU Databases..

- GPU databases offer significant improvements over the conventional CPU database when performing repetitive operations on large amounts of data. This is because GPUs can have thousands of cores and high bandwidth memory on each card.
- GPUs are typically 10x-100x faster at processing the same workloads, compared to CPUs
- GPUs are much smaller (6.5x – 20x smaller than a CPU.) Just 16 GPU-accelerated servers could perform as well as a 1000 CPU cluster
- The ability to visualize and process data in real time. Since the data is on a powerful graphics rendering engine anyway, the results are displayed at lightning speed!
Graphics Processing Unit: Programming Model
How to program a GPU?

**GPUs are programmed using the kernel programming model.**

- **Kernel:**
  - Is a simplistic program
  - Forms the basic unit of parallelism
  - Scheduled concurrently on several scalar processors in a SIMD fashion → Each kernel invocation (thread) executes the same code on its own share of the input

- **Workgroup:**
  - Logically grouping of all threads running on the same multiprocessor
Frameworks for GPU Programming

- **Compute Unified Device Architecture (CUDA):**
  - NVIDIA's Architecture for parallel computations
  - Program GPUs in CUDA C using the CUDA Toolkit

- **Open Computing Language (OpenCL):**
  - Open Standard
  - Targets parallel programming of heterogeneous systems
  - Runs on a broad range of hardware (CPUs or GPUs)
Basic Terminologies

• Host Code:
  • Executed on the CPU
  • Manages all processing on the GPU

• Device Code:
  • The kernel, is the GPU program
  • Executed massively parallel on the GPU
HOW GPU ACCELERATION WORKS

How GPU Acceleration Works

Application Code

Compute intensive operations
~10% of code

GPU

Sequential CPU code

Multi-core CPU

+
Processing Data on a GPU

1. Copy input data from CPU memory to GPU memory and allocate memory
   // cudaMemcpy((void**)&device_c, sizeof(int));
2. Load GPU program and execute,
3. Copy results from GPU memory to CPU memory

// cudaMemcpy(&c, device_c, sizeof(int), cudaMemcpyDeviceToHost);
Scale Up: Add GPUs
GPU-accelerated Database Operators

- GPUs are utilized for accelerating query processing like:
  - Relational operations
  - XML path filtering
  - Aggregation
- GPUs are as well utilized for accelerating query optimization
Figure: Classification of Co-Processing Approaches
Today’s data market - databases

• A lot of new databases are in-memory, because “memory is cheap”
**Kinetica Architecture**

**Reliable, Available and Scalable**
- Disk-based persistence
- Data replication for high availability
- Scale up and/or out

**Performance**
- GPU-accelerated (1000s cores per GPU)
- Ingest billions of records per minute
- Ultra low-latency performance from ingestion through to analytics

**Connectors**
- ODBC/JDBC
- Restful endpoints
- Open source APIs
- Native geospatial capabilities
Demo..

http://demo.kinetica.com/gaiademo/
GPU Performance Pitfalls

Data transfers between host and device:

- One of the most important performance factors in GPU programming
  - All data has to pass across the PCIe express bus
  - Bottleneck
- Limited memory capacity (1 to 16GB)
- Efficient memory management necessary
How we see the future – Hardware/Stack

• Improved programming extensions and better compilers in new CUDA/rockMwill make it easier to write good GPU code
• Faster HBM2 memory and PCilev5.0 to reduce overhead of GPU processing
• More tightly-knit hardware integration, like the Intel H-series integrated GPU processor
How we see the future of GPU databases

• The future is not just GPU databases. Different databases for different needs. The relational model is still king for most of us.

• More data = more processing power needed. Scalable database solutions that can handle growing data become more relevant

• GPUs used for compute intensive stuff, e.g. graph processing, machine learning, AI

• Rising GPU offerings in the public cloud will allow adoption by more companies
Thanks for Listening..!