Cosmos Big Data and Big Challenges

Ed Harris - Microsoft Online Services Division HTPS 2011

Outline

Introduction

- Cosmos Overview
- The Structured Streams Project
- Conclusion

What Is COSMOS?

- Petabyte Store and Computation System
 - Several hundred petabytes of data
 - Tens of thousands of computers across many datacenters
- Massively parallel processing based on Dryad
 - Similar to MapReduce but can represent arbitrary DAGs of computation
 - Automatic computation placement with data
- SCOPE (Structured Computation Optimized for Parallel Execution)
 - SQL-like language with set-oriented record and column manipulation
 - Automatically compiled and optimized for execution over Dryad
- Management of hundreds of "Virtual Clusters" for computation allocation
 - Buy your machines and give them to COSMOS
 - Guaranteed at least that many compute resources concurrently
 - May receive more resources when other VCs are under-used
- Ubiquitous access to OSD's data
 - Combining knowledge from different datasets is today's secret sauce

Cosmos and OSD Computation

- OSD Applications fall into two broad categories:
 - **<u>Back-end</u>**: Massive batch processing creates new datasets
 - **<u>Front-end</u>**: Online request processing serves up and captures information
- Cosmos provides storage and computation for Back-End Batch data analysis
 - It does not support storage and computation needs for the Front-End



COSMOS: The Service

- Data drives search and advertising
 - Web pages: Links, text, titles, etc
 - Search logs: What people searched for, what they clicked, etc
 - Browse logs: What sites people visit, the browsing order, etc
 - Advertising logs: What ads do people click on, what was shown, etc
- We ingest or generate a couple of PiB every day
 - Bing, MSN, Hotmail, Client telemetry
 - Web crawl snapshots
 - Structured data feeds
 - Long tail of other data sets of interest
- COSMOS is the backbone for Bing analysis and relevance
 - Click-stream information is imported from many sources and "cooked"
 - Queries analyzing user context, click commands, and success are processed
- COSMOS is a service
 - We run the code ourselves (on many tens of thousands of servers)
 - Users simply feed in data, submit jobs, and extract the results

Outline

- Introduction
- Cosmos Overview
- The Structured Streams Project
- Conclusion

Cosmos Architecture from 100,000 Feet

SCOPE Layer:

- -- SCOPE Code is submitted to the SCOPE Compiler
- -- The optimizer make decisions about execution plan and parallelism
- Algebra (describing the job) is built to run on the SCOPE Runtime

Execution Layer:

- -- Jobs queues up per Virtual Cluster
- -- When a job starts, it gets a Job Mgr to deploy work in parallel close to its data
- -- Many Processing Nodes (PNs) host execution vertices running SCOPE code

Store Layer:

- -- Many Extent Nodes store and compress replicated extents on disk
- -- Extents are combined to make unstructured streams
- -- CSM (COSMOS Store Manager) handles names, streams, & replication



The Store Layer

- Extent Nodes:
 - Implement a file system holding extents
 - Each extent is up to 2GB
 - Compression and fault detection are important parts of the EN
- CSM: COSMOS Store Manager
 - Instructs replication across 3 different ENs per extent
 - Manages composition of streams out of extents
 - Manages the namespace of streams



The Execution Engine

- Execution Engine:
 - Takes the plan for the parallel execution of a SCOPE job and finds computers to perform the work
 - Responsible for the placement of the computation close to the data it reads
 - Ensures all the inputs for the computation are available before firing it up
 - Responsible for failures and restarts
- Dryad is similar to Map-Reduce



The SCOPE Language

- SCOPE (Structured Computation Optimized for Parallel Execution)
 - Heavily influenced by SQL and relational algebra
 - Changed to deal with input and output streams



- SCOPE is a high level declarative language for data manipulation
 - It translates very naturally into parallel computation

The SCOPE Compiler and Optimizer

- The SCOPE Compiler and Optimizer take SCOPE programs and create:
 - The algebra describing the computation
 - The breakdown of the work into processing units
 - The description of the inputs and outputs from the processing units
- Many decisions about compiling and optimizing are driven by data size and minimizing data movement



The Virtual Cluster

- Virtual Cluster: a management tool ٠
 - Allocates resources across groups within OSD
 - Cost model captured in a queue of work (with priority) within the VC
- Each Virtual Cluster has a guaranteed capacity ٠
 - We will bump other users of the VC's capacity if necessary
 - The VC can use other idle capacity

100 Hi-Pri PNs	500 Hi-Pri PNs	20 Hi-Pri PNs	1000 Hi-Pri PNs	350 Hi-Pri PNs
Work Queue	Work Queue	Work Queue	Work Queue	Work Queue
VC-A	VC-B	VC-C	VC-D	VC-E

Outline

- Introduction
- Cosmos Overview
- The Structured Streams Project
- Conclusion

Introducing Structured Streams

- Cosmos originally supported streams
 - An unstructured byte stream of data
 - Created by append-only writing to the end of the stream
- Structured streams are streams with metadata
 - Metadata defines column structure and affinity/clustering information
- Structured streams simplify extractors and outputters
 - A structured stream may be imported into scope without an extractor
- Structured streams offer performance improvements
 - Column features allow for processing optimizations
 - Affinity management can dramatically improve performance



Today's Use of Extractors and Outputters



Metadata, Streams, Extractors, & Outputters

- Scope has metadata for the data it is processing
 - Extractors provide metadata info as they suck up unstructured streams
- Processing the Scope queries ensures metadata is preserved
 - The new results may have different metadata than the old
 - Scope knows the new metadata
- Scope writes structured streams
 - The internal information used by Scope is written out as metadata
- Scope reads structured streams
 - Reading a structured stream allows later jobs to see the metadata

The Representation of a Structured Stream on Disk Is Only Visible to Scope!



Streams, Metadata, and Increased Performance

- By adding metadata (describing the stream) *into* the stream, we can provide performance improvements:
 - <u>Cluster-Key access</u>: random reads of records identified by key
 - <u>Partitioning and affinity</u>: data to be processed together (sometimes across multiple streams), can be placed together for faster processing
- Metadata for a structured stream is kept <u>inside</u> the stream
 - The stream is a self-contained unit
 - The structured stream is still an unstructured stream (plus some stuff)



Cluster-Key Lookup

- Cluster-Key Indices make a huge performance improvement
 - Today: If you want a few records, you must process the whole stream
 - Structured Streams: Directly access the records by cluster-key index
- How it works:
 - Cluster-Key lookup is implemented by having indexing information contained in the metadata inside the stream
 - The records must be stored in cluster-key order to use cluster-key lookup
 - Cosmos managed index generated at structured stream creation



Implementing Partitioning and Affinity

- Joins across streams can be very expensive
 - Network traffic is a major expense when joining large datasets together
 - Placing related data together can dramatically reduce processing cost
- We affinitize data when we believe it is likely to be processed together
 - Affinitization places the data close together
 - If we want affinity, we create a "partition" as we create a structured stream
 - A partition is a subset of the stream intended to be affinitized together



Case Study 1: Aggregation



across network

Case Study 2: Selection

SELECT URL, feature1, feature2 FROM Web-Table WHERE URL == www.imdb.com;



Unstructured Datasets

	Partition	Range	Metadata
Partition			
Metadata	P100	www.imc.com ⇔ www.imovie.com	
	P101	www.imz.com	



Structured Datasets (Sstream) (partitioned by URL, sorted by URL)

Case Study 3: Join Multiple Datasets



across network

Outline

- Introduction
- Cosmos Overview
- The Structured Streams Project
- Conclusion

Takeaways

- Cosmos manages OSD's (Online Services Division) data and computation
 - Hundreds of petabytes of data
 - Many thousands of very large batch jobs per day
 - Tens of thousands of servers in multiple datacenters
- All the data in the same addressable store is *essential* to the business
 - The money lies in combining data in surprising ways...
- Cosmos is evolving:
 - Structured Streams combines large scale parallelism with DB techniques to dramatically improve the efficiencies of computation
- In the works:
 - Integrated random and sequential processing over the same data
 - Reliable workflow via pub-sub
 - Transactionally correct changes over many petabytes of data

Cosmos needs fresh blood!

 Send your students our way! Bing sponsors 60+ PhD and undergraduate internships per year – <u>edharris@microsoft.com</u>