Enabling Edge Analytics for Remote Data

Michael Hummel, CTO & Director Cisco ParStream
BRKDEV-1005
Agenda

- Introduction
- IoT Analytics Pattern
- IoT Analytics needs Edge and Fog Computing
- Cisco’s Edge Analytics Fabric
- Cisco ParStream’s Architecture
- Cisco ParStream’s Key Features
- Conclusion
IoT Analytics Delivers
15% More Output from Renewable Energy Sources

30TB
Analyze Data in Real-time

15%
Increase Efficiency

$158M/yr
Generate Operational/Economic Benefits

(20,000 Wind Turbines; 10 GW Capacity; .3 Capacity Factor; $40/MW-hour)
Agenda

• Introduction

• IoT Analytics Pattern

• IoT Analytics needs Edge and Fog Computing

• Cisco’s Edge Analytics Fabric

• Cisco ParStream’s Architecture

• Cisco ParStream’s Key Features

• Conclusion
What is an IoT Project?

Leveraging Machine Generated Data and Networking for Business Benefit
Common Themes (IoT Business Objectives)

Frequent Goals

- **Remote Monitoring** (Oil & Gas Drilling, Oil & Gas Refineries, Wind Turbines)
- **Preventive Maintenance** (Factory Robots, Airplane Tires, Energy Grid)
- **Personnel Safety** (Refinery Gas Leaks, Contamination Containment)
- **Real-time Quality Assessment** (Oil Drilling, Manufacturing Cell, Train Repair)
- **Asset Health** (Readiness Assessment)
- **Efficiency Through Digitization & Automation** (Smart Meters -> Utility Billing, etc.)
- **Cost Reduction Through Better Facilities Management** (Energy Management & Reduction)
IOT is a Journey – Start Small, Grow Fast

Business Focus

1st IoT Project
- Discovery
- Pattern Recognition

2nd IoT Project
- Discovery
- Pattern Recognition

Business Goal

Data Source

IoT Data Sources

1st Business Benefits

2nd Business Benefits

more IoT Project
A typical IoT Project

### Discovery

- Find reoccurring pattern
- Analyze causes
- Define action
- Analyze effect

### Pattern Recognition

- Measure pattern
- pattern + action = rule
- Run rules continuously

#### Business Goal

- Define action

#### Data Sources

- Measure pattern

#### Business Benefits

- Accelerate decision-making
Agenda

• Introduction
• IoT Analytics Pattern
• IoT Analytics needs Edge and Fog Computing
• Cisco’s Edge Analytics Fabric
• Cisco ParStream’s Architecture
• Cisco ParStream’s Key Features
• Conclusion
Delivering actionable insights requires use-case specific applications & data integration

**M2M pattern – siloed vertical solutions**

- Network Analytics
- POS Analytics

- No Data Integration
- Individual Solutions
- Dedicated Ressources

**Big Data pattern – horizontal integration**

- Client: Network Analytics, POS Analytics, Fraud Analytics
- Cloud / Data Center
- Network
- Source: POS, POS, POS, POS

4 billion records

© 2016 Cisco and/or its affiliates. All rights reserved. Cisco Public
Huge Amounts Of Time-Critical Dispersed Data Requires Decentralization To Be Added To The Mix

**Big Data pattern – horizontal integration**

- **Client**
  - Network Analytics
  - POS Analytics
  - Fraud Analytics
  - ?

- **Cloud / Data Center**
  - Centralized Processing

- **Network**
  - Shared WAN

- **Source**
  - 50 billion devices

**IoT pattern – de-centralization at scale**

- **Client**
  - Network Analytics
  - POS Analytics
  - Fraud Analytics
  - ?

- **Cloud / Data Center**
  - Centralized Processing

- **Fog**

- **Edge**

- **Source**
  - 50 billion devices

---

© 2016 Cisco and/or its affiliates. All rights reserved. Cisco Public
IOT Analytics Needs a New Approach

Traditional Approach – Taking Data to the Processing

- Cloud
- Big Data
- Analytics
- Applications

New Paradigm – Taking Processing to the IoT Data

- IoT Device
- Edge Node
- Fog Node
- Cloud

IoT Computing is a New Computing Paradigm
IOT Analytics Needs Multi-Tier Flexibility

2-Tier
- IoT Device
- Data
- Cloud

3-Tier
- Factory Device
- Data
- Factory
- Edge Node
- Data Center

4-Tier
- Bore Head
- Data
- Drill Tower
- Data
- Oil Rig
- Fog Node
- Data Center

http://www.kgs.ku.edu/Publications/Oil/primer12.html
Customer Example - Global Energy Company

Product Quality Assessment at the Well - 7,000 Oil Wells x 1TB Data/Day/Well

- IT Central
- Data Center
- Fog Processing - 300 Platforms
- Edge Processing - 7,000 Oil Wells
Fog Computing Patterns (vertical market examples)

**Pattern**
- Headquarters
  - IT Central
- Regional
  - Data Center
- Processing Platforms
  - Fog Processing
- Edge Location
  - Sensors & Edge Processing

**Manufacturing**
- WW Operations
  - Data Center / Cloud
  - Data Mining & Modeling
- Factory
  - Fog Aggregation Nodes
  - Factory History & Trends
  - Reports & Dashboards
- Plant Areas
  - Fog Node
  - Real-Time Dashboards
  - Predictions
- Robots & Machines
  - Sensors, Alerting,
  & Edge Analytics

**MNOs**
- Customer Experience
  - Data Center / Cloud
  - Analysis of customer behavior
  vs network performance
- Mobile Network Operations
  - Fog Aggregation Nodes
  - Network Performance Monitoring
  - Global incident investigation
- Cell Tower
  - Fog Node
  - Alerting
  - Local analytics & predictions
- Mobile Device
  - Sensors
  - Alarms
  - Edge Analytics

and many more…
The Case for Edge and Fog Computing

1. Too much data to move it all into the cloud… need filtering, aggregation, batching etc needed
2. Network connectivity is not always available… need local intelligence without cloud connectivity
3. Network bandwidth is limited… need to analyze all data but de-centrally
4. Response time is of essence… need to analyze data as soon as possible.
5. Data is in the wrong or in different formats… need to convert / harmonize with local context.
6. Data required reliable time stamp… stamp before sending, before buffering
IoT World Forum Reference Model
Specifically Includes Edge + Fog Computing

Levels

7. Collaboration & Processes
   (Involving People & Business Processes)

6. Application
   (Reporting, Analytics, Control)

5. Data Abstraction
   (Aggregation & Access)

4. Data Accumulation
   (Storage)

3. Edge Computing
   (Data Element Analysis & Transformation)

2. Connectivity
   (Communication & Processing Units)

1. Physical Devices & Controllers
   (The “Things” in IoT)
Agenda

• Introduction
• IoT Analytics Pattern
• IoT Analytics needs Edge and Fog Computing
• Cisco’s Edge Analytics Fabric
• Cisco ParStream’s Architecture
• Cisco ParStream’s Key Features
• Conclusion
Cisco’s EAF – Edge Analytics Fabric
All you need for Edge + Fog Computing of IoT Data
EAF’s Open, Modular Architecture

Monitor data stream, store all raw data long-term

Edge Analytics Fabric (EAF)
- Sensor
  - Data Capture
- Cloud
  - Applications Services
- Connected Streaming Analytics
- Cisco ParStream Database

Monitor data stream, only store selected data long-term

Edge Analytics Fabric (EAF)
- Sensor
  - Data Capture
- Cloud
  - Applications Services
- EAF Message Broker
- Connected Streaming Analytics
- Cisco ParStream Database
EAF - An Open System Enabling MicroServices

Micro-Services

- Event Stream Processing
- Correlation
- Aggregation
- Filtering
- Archiving
- Machine Learning
- Access & Integration (CIS)
- Analytics (Data Prep)

Edge Analytics Fabric

- Connected Streaming Analytics
- EAF Message Broker

Cisco ParStream Database

Device or Controller

- Generating Data
- Capturing Data
- Aggregating Data
- Leveraging Data
- Analyzing Data

BRKDEV-1005
Sample Topology

**Edge Location**
Sensors & Edge Processing

**Fog Location**
Fog Processing

**Data Center**
Central Processing

---

**Broker**

**Flow Link**

**Temp Link**

---

**Broker**

**Flow Link**

**Temp. Link**

---

**Broker**

**Link**

**Connected Streaming Analytics**

---

**Broker**

**Link**

**Cisco ParStream**

---

**Broker**

**Link**

**Dashboard**

**Data Virtualization**

---

**Broker**

**Link**

**Data Lake**

---

Cisco live!
Data Flow Editor
Data Transformation and Enhancement
Event Stream Processing

Connected Streaming Analytics

- Analytics at the Edge
- Operates on streaming data
- Compares current data with historic data
- Compares multiple simultaneous streams
- Complex statistical analysis
- Runs in a router or switch (small footprint)
Historian Database

ParStream

- SQL Compliant
- > 1M inserts/second
- < 1 sec response times for tables with 1B rows
- Handles structured and semi-structured data
- Geo-distributed queries
- Automatically connected to the Fabric
Edge Analytic Fabric Monitor
Administration and Configuration
Agenda

- Introduction
- IoT Analytics Pattern
- IoT Analytics needs Edge and Fog Computing
- Cisco’s Edge Analytics Fabric
- Cisco ParStream’s Architecture
- Cisco ParStream’s Key Features
- Conclusion
Cisco ParStream
Analytical IoT Data Historian in Cisco’s Edge Analytics Fabric

- Analytical Database for IoT Data
- Columnar + Key-Value + Compressed Bitmap Index
- MPP + Scale-out + Geo-Distributed
- Bulk + Streaming
- High Availability Cluster
- SQL compatible
- FOG ready
- Built from Scratch in C++
Choose Your Database Based on Use-Case

- **Stream-Analytics**
  - Complex Event Processing
  - In-Memory DB

- **Operations Analytics**
  - OLTP Reporting
  - OLAP

- **Batch-Analytics**
  - Cassandra
  - Hadoop

- **Real-Time IoT Analytics**
  - Cisco ParStream
  - Massively Parallel (MPP)
  - Real-Time

**Big Data**
- Gigabyte
- Terabyte
- Petabyte

**Response Time**
- Fast
- Slow

**Key Terms**
- Big Data
- Massively Parallel (MPP)
- Real-Time
- OLTP
- Reporting
- OLAP
- Cassandra
- Hadoop
- Gigabyte
- Terabyte
- Petabyte
- Stream-Analytics
- Operations Analytics
- Batch-Analytics
- Real-Time IoT Analytics
- Cisco ParStream
## Databases for IoT Data Discovery

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Product</th>
<th>Cisco ParStream</th>
<th>Columnar Databases</th>
<th>Row-Databases</th>
<th>KeyValue Stores</th>
<th>Hadoop Batch</th>
<th>Hadoop Streaming</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume</td>
<td></td>
<td>●</td>
<td>□</td>
<td></td>
<td>□</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Velocity</td>
<td></td>
<td>□</td>
<td>□</td>
<td></td>
<td>□</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Variety</td>
<td></td>
<td>□</td>
<td>□</td>
<td></td>
<td>□</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Algorithmic locality</td>
<td></td>
<td>●</td>
<td>□</td>
<td></td>
<td>□</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Latency</td>
<td></td>
<td>□</td>
<td>□</td>
<td></td>
<td>□</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Query Throughput</td>
<td></td>
<td>●</td>
<td>□</td>
<td></td>
<td>□</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Schema</td>
<td></td>
<td>□</td>
<td>□</td>
<td></td>
<td>□</td>
<td>□</td>
<td>□</td>
</tr>
<tr>
<td>Selectivity</td>
<td>Selected</td>
<td>□</td>
<td>Lookup</td>
<td>Lookup</td>
<td>All Data</td>
<td>All Data</td>
<td></td>
</tr>
</tbody>
</table>

- **Good**
- **Fair**
- **Poor**
Concurrent Import and Analytics

Lockless architecture enables continuous micro-bulk import and high-speed read-only query processing.
ParStream’s Patented Technology Provides a Competitive Advantage

1. **High Performance Compressed Indexes**
   Provide ultra-high query performance

2. **Massive Parallel Processing**
   Delivers linear scalability and high query throughput

3. **Lockless Architecture**
   Enables ultra-fast query and data import performance

4. **Small Footprint**
   Enables analytics at the edge with a low TCO
ParStream’s Patented High Performance Compressed Index Enables Massive Performance Gain

Standard Index Architecture

- Decompression latency
- High memory and CPU load
- Not suitable for Big Data
- Limited query variability and throughput

ParStream’s Superior Index Architecture

+ Zero latency to start analytics
+ No decompression or full table scans
+ Massively reduced memory and IO load
+ Ultra-high throughput
Agenda

• Introduction
• IoT Analytics Pattern
• IoT Analytics needs Edge and Fog Computing
• Cisco’s Edge Analytics Fabric
• Cisco ParStream’s Architecture
• Cisco ParStream’s Key Features
• Conclusion
Demo
ParStream Has Super Fast Query Response Times Even on Billion of Records!

Demo
Cisco ParStream’s Key Features

- Parallel Query Execution Engine
- Bitmap Operations
- Dynamic Column
- Geo-Distributed Analytics
- Advanced Analytics / Machine Learning
MPP Execution Engine
Designed for Modern Processor Architectures

• 3 Element Execution Model – Nodes, Queues, and Resource Pools
• Query is parsed, parameterized, and enriched using metadata
• Compiled into a Execution Tree
  • Each execution tree node processes blocks of data
  • Data is transferred between nodes via queues
• Optimizer assigns plan steps to available hardware resources from resource pools

![Execution Tree Diagram]
MPP Query Optimization

- Maximized query parallelization and distribution
- Extensive use of bitmap index
  - Filtering
  - Pure Bitmap Aggregation
  - Cost Estimation
- Partition specific selection of data and index operation
- All processor cores of all servers of a cluster are engaged in query execution; this scales out linearly

SMP Parallel Processing

- Multiple query nodes
- Nodes processes independent data blocks
- Data blocks pipelined through query tree

MPP Parallel Processing

- Sub-trees processing independent partitions
- Optimizer decides upon breath and depth of execution
Cisco ParStream’s Key Features

• Parallel Query Execution Engine
• Bitmap Operations
• Dynamic Column
• Geo-Distributed Analytics
• Advanced Analytics / Machine Learning
Bitmap Index – Basic Concept

**Data Table ‘t_sales’**

<table>
<thead>
<tr>
<th>ROWID</th>
<th>Color</th>
<th>Size</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>brown</td>
<td>42</td>
<td>100.00</td>
</tr>
<tr>
<td>2</td>
<td>black</td>
<td>42</td>
<td>133.50</td>
</tr>
<tr>
<td>3</td>
<td>brown</td>
<td>45</td>
<td>130.20</td>
</tr>
<tr>
<td>4</td>
<td>red</td>
<td>42</td>
<td>105.99</td>
</tr>
<tr>
<td>5</td>
<td>black</td>
<td>41</td>
<td>130.00</td>
</tr>
</tbody>
</table>

**Bitmap Color**

<table>
<thead>
<tr>
<th></th>
<th>brown</th>
<th>black</th>
<th>red</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Bitmap Size**

<table>
<thead>
<tr>
<th></th>
<th>41</th>
<th>42</th>
<th>45</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Query**

```sql
SELECT COUNT(*)
FROM t_sales
WHERE color = 'brown'
AND size IN (42, 45)
```

**1st Step: Fetch and filter index**

<table>
<thead>
<tr>
<th>Color = brown</th>
<th>Size in (42, 45)</th>
<th>Bitmap index result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brown AND (</td>
<td>42 OR 45)</td>
<td>count(*)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**2nd Step: Count ‘1’**

`count(*) 2`
Cisco ParStream’s High Performance Compressed Index

Supported Bitmap Types

• **Equal Index**
  - Small number of discrete values, <1 million
  - Typically used for integer & char

• **Range Index**
  - Similar to equal index but coding optimized for range queries
  - Typically used for date, time, time-stamps

• **Binned Index**
  - Two or more values are binned into one index. The distribution of values into one index can be equally distributed or custom defined.
  - Used for >1 million different values, e.g. floating point values
Cisco ParStream’s High Performance Compressed Index

Bitmap Usage

• **WHERE**
  • Single and multi-column combinatorial filtering
  • All operators including equal, larger / smaller
  • WHERE IN with thousands of values
  • LIKE on text triplets

• **GROUP BY**
  • Super fast for up to thousands of groups per partition
  • Automatic switch to value based GROUP BY for larger group counts

• **COUNT, COUNT DISTINCT, SUM, AVG...**
  • Aggregation functions
Cisco ParStream’s Key Features

- Parallel Query Execution Engine
- Bitmap Operations
- Dynamic Column
- Geo-Distributed Analytics
- Advanced Analytics / Machine Learning
IoT Data – Two Frequently Applied Solution Patterns

Time-Synchronized IoT data

Sensor 1
Sensor 2
Sensor 3
Timestamp
12:00:06 12:00:07 12:00:08 12:00:09 12:00:10

Event-Stream IoT data

Sensor 1
Sensor 2
Sensor 3
Timestamp
12:00:06 12:00:07 12:00:08 12:00:09 12:00:10

Stored in Columnar structure

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Sensor 1</th>
<th>Sensor 2</th>
<th>Sensor 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>12:00:06</td>
<td>100</td>
<td>43</td>
<td>25</td>
</tr>
<tr>
<td>12:00:07</td>
<td>90</td>
<td>41</td>
<td>28</td>
</tr>
<tr>
<td>12:00:08</td>
<td>88</td>
<td>11</td>
<td>105</td>
</tr>
<tr>
<td>12:00:09</td>
<td>88</td>
<td>50</td>
<td>110</td>
</tr>
</tbody>
</table>

Stored in Key-Value structure

<table>
<thead>
<tr>
<th>Key</th>
<th>Timestamp</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor 1</td>
<td>12:00:06</td>
<td>100</td>
</tr>
<tr>
<td>Sensor 3</td>
<td>12:00:06</td>
<td>25</td>
</tr>
<tr>
<td>Sensor 2</td>
<td>12:00:06:510</td>
<td>43</td>
</tr>
<tr>
<td>Sensor 1</td>
<td>12:00:07</td>
<td>93</td>
</tr>
<tr>
<td>Sensor 3</td>
<td>12:00:07</td>
<td>28</td>
</tr>
</tbody>
</table>

Schema-on-write
Needs careful preparation, enables easy and super fast analytics

Schema-on-read
Needs no preparation, stores everything, analytics complex and slow
ParStream’s Dynamic Column Feature Delivery Both

User sees dynamically created columnar representation

ParStream hides complexity of key-value like storage

```
SELECT Timestamp, OilTemp, PowerProduced, Temp3
FROM t_dyncol
WHERE ...
```

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>OilTemp</th>
<th>Power Produced</th>
<th>Temp3</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015-07-15 12:00:00</td>
<td>67</td>
<td>56</td>
<td>NULL</td>
</tr>
<tr>
<td>2015-07-15 12:00:01</td>
<td>...</td>
<td>...</td>
<td>NULL</td>
</tr>
<tr>
<td>2015-07-17 12:00:00</td>
<td>68</td>
<td>57</td>
<td>34</td>
</tr>
</tbody>
</table>

**Key**

<table>
<thead>
<tr>
<th>Key</th>
<th>Timestamp</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temp1</td>
<td>2015-07-15 12:00:00</td>
<td>26</td>
</tr>
<tr>
<td>WindDir</td>
<td>2015-07-15 12:00:00</td>
<td>153</td>
</tr>
<tr>
<td>Humidity</td>
<td>2015-07-15 12:00:00</td>
<td>42</td>
</tr>
<tr>
<td>Power Produced</td>
<td>2015-07-15 12:00:00</td>
<td>56</td>
</tr>
<tr>
<td>OilTemp</td>
<td>2015-07-15 12:00:00</td>
<td>67</td>
</tr>
<tr>
<td>Temp2</td>
<td>2015-07-15 12:00:01</td>
<td>26</td>
</tr>
<tr>
<td>Temp1</td>
<td>2015-07-15 12:00:07</td>
<td>17</td>
</tr>
<tr>
<td>Temp3</td>
<td>2015-07-17 12:00:00</td>
<td>34</td>
</tr>
<tr>
<td>NewSensor</td>
<td>2015-07-17 12:00:00</td>
<td>34</td>
</tr>
</tbody>
</table>

**Flexible Schema-on-read**

No need to extend schema when new sensors are added

**Ultra-fast Schema-on-read**

Due to ParStream’s unique high-performance-compressed-indexes

Cisco live!
```sql
SELECT
    SUM(A.val_uint64) AS avg_sensor_1,
    SUM(B.val_uint64) AS avg_sensor_2
FROM
    t_sensor_dyncol A
INNER JOIN
    t_sensor_dyncol B
ON
    A.ts = B.ts
    AND A.sensor_id = 1
    AND B.sensor_id = 2
WHERE A.ts BETWEEN (...)
GROUP BY ts_day
ORDER BY ts_day
;

CREATE TABLE t_sensor_dyncol (  
    ts TIMESTAMP SEPARATED BY ts_day INDEX RANGE  
    ,sensor_id UINT64 INDEX EQUAL  
    ,ts_day TIMESTAMP INDEX RANGE  
    ,val_name VARSTRING(256) COMPRESSION  
    ,hash64 MAPPING_LEVEL PARTITION INDEX EQUAL  
    ,val_uint64 UINT64 INDEX EQUAL  
)  
PARTITION BY ts_day  
DISTRIBUTE OVER ts_day WITH REDUNDANCY 1  
ORDER BY ts, sensor_id
;
```

Cisco ParStream’s Key Features

- Parallel Query Execution Engine
- Bitmap Operations
- Dynamic Column
- Geo-Distributed Analytics
- Advanced Analytics / Machine Learning
Edge Analytics Delivers Real-time Insights by Minimizing Network Traffic

Overcoming Bandwidth Limitations and Reducing Report Delays Requires Analytics to be Pushed Closer to the Data Source

Central Analytics Intelligence

Application

Query

Search Results
40 Records Found

Database

More Than 20 Billion
Records Returned

4 Billion Records

4 Billion Records

4 Billion Records

4 Billion Records

Hybrid Edge / Centralized Intelligence

Application

Query

Search Results
40 Records Found

ParStream Geo-Distributed Server

<100 Records

7 Records

18 Records

5 Records

12 Records

8 Records

ParStream

ParStream

ParStream

ParStream

ParStream

ParStream

ParStream

ParStream

ParStream
## GDA Architecture vs. MPP vs. Federated

GDA is a Loosely Coupled Cluster with Remote Query Execution, Applying Multi-level Optimization and Selective Query Distribution

<table>
<thead>
<tr>
<th></th>
<th>Analytical Cluster (MPP)</th>
<th>Geo-Distributed Deployment</th>
<th>Federation Servers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Architecture</strong></td>
<td>Tightly Coupled Nodes</td>
<td>Loosely Coupled Nodes</td>
<td>Independent Sub-systems</td>
</tr>
<tr>
<td><strong>High Availability</strong></td>
<td>&quot;System&quot; Redundancy</td>
<td>&quot;Node&quot; Redundancy</td>
<td>&quot;Sub-system&quot; Redundancy</td>
</tr>
<tr>
<td></td>
<td>Full or No Results</td>
<td>Full or partial results</td>
<td>Full or No Results</td>
</tr>
<tr>
<td><strong>Data Model</strong></td>
<td>One Model</td>
<td>One Model with Local</td>
<td>Joint View on</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extensions</td>
<td>Local Models</td>
</tr>
<tr>
<td><strong>Query Execution</strong></td>
<td>Data-structure-aware</td>
<td>Data-structure-aware</td>
<td>Data-structure-aware</td>
</tr>
<tr>
<td></td>
<td>system optimization</td>
<td>execution plan with</td>
<td>execution-plan-translation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data-aware local</td>
<td>with independent execution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>optimization</td>
<td></td>
</tr>
</tbody>
</table>
Cisco ParStream’s Key Features

- Parallel Query Execution Engine
- Bitmap Operations
- Dynamic Column
- Geo-Distributed Analytics
- Advanced Analytics / Machine Learning
User Defined Functionality

Output Parameter Type

<table>
<thead>
<tr>
<th>Scalar</th>
<th>Set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Scalar</td>
<td>User Defined Scalar Function</td>
</tr>
<tr>
<td>Set</td>
<td>User Defined Aggregate Function</td>
</tr>
</tbody>
</table>

UDA

“External” UDTO aka “R-Integration”
Extension-Points
C++ API Allows Different User Extension

UDS
- Sort
- Agg
- Agg
- Calc
- Filter
- Fetch

UDA
- Sort
- Agg
- Agg
- Agg
- Calc
- Filter
- Fetch

UDT
- Sort
- Union
- UDT
- Calc
- Filter
- Fetch
Advanced Analytics / Machine Learning Interface

- Extension of analytical scope
- Utilizes external data processors
  - R, Python, ETL, EMB, ruby, Java, C…
- On-database cluster integration

Named xUDTO…

*External User-Defined-Table-Operator*
Code Example – Execution of the R-Script - illustrative

Any BI Tool could trigger this query as it is pure SQL!

```sql
SELECT * FROM anomalyDetection (
    ON (
        SELECT
            sensor_name
            ,ts
            ,value
        FROM t_sensor_data
        WHERE ts > '2015-01-01 00:00:00'
    )
    PARTITION BY sensor_name
    ORDER BY ts ASC
    RETURNS (
        sensor_name VARSTRING
        ,ts TIMESTAMP
        ,anomalyScore DOUBLE
    )
    COMMAND_LINE_ARGS(
        'saxAlphabetSize=5
        anomalyThreshold=0.45'
    )
)
```

- **The name of the function or xUDTO**
- **Could be a parameter defined in the BI Tool**
- **The function is invoked in parallel for each subset of data per “sensor_name”**
- **Ordering will be calculated by the ParStream SQL engine, not by the external engine!**
- **Return schema defined “on-the-fly”**
- **Pass arguments to the R-Script, e.g. thresholds etc. at runtime on the fly!**
Code Example - Simple R-Script – Focus on the “Glue Code”

#reading rows from ParStream via STDIN

f <- FILE("stdin","r")

d <- READ.CSV (f, header = FALSE, sep = ";", quote = "\\", dec = ".", fill = TRUE, comment.char = "")

CLOSE(f)

#Do something with the rows from the database of that particular data partition

#Write the calculated result to STDOUT so that ParStream can pick it up for further processing
#Here we just echo back the input unchanged

WRITE.TABLE (d, file = stdout(),row.names = FALSE, na = "", col.names=FALSE, sep=";")
Agenda

• Introduction
• IoT Analytics Pattern
• IoT Analytics needs Edge and Fog Computing
• Cisco’s Edge Analytics Fabric
• Cisco ParStream’s Architecture
• Cisco ParStream’s Key Features
• Conclusion
IoT Analytics Is Living On The Network

- **IoT** = Leveraging machine generated data and networking for business benefit

- **IoT is a journey** – start small, grow fast – humans + IT hand-in-hand

- **IoT analytics needs a new approach** – moving processing to data, multi-tier

- **Cisco’s Edge-Analytics-Fabric (EAF)** is living on the network, adds decentral data processing to the solution stack, is open and modular, enables MicroServices to run on Edge + Fog Nodes

- **Cisco ParStream is the next generation data historian**, super fast insert and analytical querying – supports SQL, AA + ML and runs decentrally on the network
Complete Your Online Session Evaluation

• Give us your feedback to be entered into a Daily Survey Drawing. A daily winner will receive a $750 Amazon gift card.

• Complete your session surveys through the Cisco Live mobile app or from the Session Catalog on CiscoLive.com/us.

Don’t forget: Cisco Live sessions will be available for viewing on-demand after the event at CiscoLive.com/Online
Continue Your Education

• Demos in the Cisco campus
• Walk-in Self-Paced Labs
• Lunch & Learn
• Meet the Engineer 1:1 meetings
• Related sessions
Thank you
Backup
Fog Computing Pattern In IoT

When to consider FOG computing
• Data is created de-centrally
• Big Data Volume > 1 billion data points
• Deriving insights close in time is an advantage for the business
• Network bandwidth insufficient or too expensive (e.g. satellite link)

Benefits to the customer
• Availability of all data – big volumes with full granularity
• Very short report delay (a few seconds instead of hours / days)
• Compliance with local data storage requirements (e.g. safe harbor)
• Highly cost efficient network usage

Benefits to Cisco
• Unique offering both from a functional and financial perspective
• Uplifts network hardware product portfolio (routers, switches…)

Headquarters
IT Central

Regional
Data Center

Processing
Platforms
Fog Processing

Edge Location
Sensors & Edge Processing
## Choose Your Database Based On Data Access Pattern

<table>
<thead>
<tr>
<th></th>
<th>Hadoop</th>
<th>Columnar Database</th>
<th>Key-Value-Store</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use-Case Focus</strong></td>
<td>Analytical processing of all Data, ETL-style</td>
<td>Ad-hoc analytics of flexibly selected Data</td>
<td>Lookup Data</td>
</tr>
<tr>
<td></td>
<td>All processing done in memory, results flushed to disk</td>
<td>Mainly in-memory caching of hot data, small process cash for temporary results</td>
<td>No data processing</td>
</tr>
<tr>
<td><strong>Read-Access</strong></td>
<td>Sequential read from a few files</td>
<td>Sequential read from many files concurrently (100 to 10,000 files)</td>
<td>Massive number of lookups, i.e. highly selective reads of single value identified by hash-key</td>
</tr>
<tr>
<td><strong>Write-Access</strong></td>
<td>Sequential write in large junkes (entire rows are stored together in large text or binary files)</td>
<td>Sequential write on many files in parallel (one file per column)</td>
<td>Massive number of small writes for key-value pairs</td>
</tr>
<tr>
<td><strong>Storage Architecture</strong></td>
<td>Dedicated storage cluster consisting of 5 to 100 servers</td>
<td>Cluster nodes consist of strong processing unit with several multi-core chipsets and strong local storage with HW load balancers</td>
<td>Dedicated storage cluster consisting of 5 to 100 servers</td>
</tr>
<tr>
<td></td>
<td>Information of a row typically stays together as data is often schemaless</td>
<td>Logical data sharding to maximize data local processing.</td>
<td>Physical data sharding to evenly distribute write and read workload on many servers to increase throughput.</td>
</tr>
<tr>
<td></td>
<td>Small number of partitions translates in small number of directories / files (&lt;&lt; 1 million)</td>
<td>Large number of partitions to enable selective read. Large number of directories / files (&gt; 1 millions)</td>
<td></td>
</tr>
<tr>
<td><strong>Data organization</strong></td>
<td>File or data block level async write to different storage cluster member</td>
<td>Data redundancy through copies of logical partitions on more than one cluster node</td>
<td>Redundancy through storing multiple copies of key-value pairs on different cluster members</td>
</tr>
</tbody>
</table>
Background Concepts

- **Everything is a Node**

- **Brokers**: Enable communications between nodes. Publish-subscribe and (bi-directional) message exchange. Connect to other brokers to form a graph. Provide introspection capabilities.

- **Links**: Mechanism to get data in/out of Node. Containers for nodes. Device, Bridge, Engine, DataFlow, DQL, System, JDBC links. Implemented via node API. Created using the various sdks. Every link is microservice.

- **Edges**: are primarily small compute environments responsible for connecting to, aggregating data from, and controlling “devices”. Broker(s) and links are deployed to edges. Cisco Routers/Gateways.

- **Fog/Aggregation**: Aggregate data from edges. Historian Databases are hosted here. Used for open ended analytics based on detailed history.

- **Data Center Cloud**: Large scale compute center. Put data in the data lake for large scale, offline processing.