Big Data @ Microsoft
A View from CISL

Raghu Ramakrishnan
Outline

• Big Data
  – The Digital Shoebox

• Selected topics:
  – Tiered Storage
  – Compute Fabric
    • REEF and YARN
Cloud Information Services Lab (CISL)

• Applied research for Cloud and Enterprise (CE)
• Focus areas:
  – Cloud data platforms and data-driven next-gen enterprise solutions
• Modus innovatii:
  – Embedded with the product team
  – Vehicle to engage deeply with MSR
  – Work with partners to apply ML/Cloud technology
Big Data
What’s the big deal?
What’s New?

• **What we’re doing with it!**
  – The tech is best thought of in terms of what it enables

• **Why is this more than tech evolution?**
  – Cloud services + advances in analytics + HW trends = Ability to cost-effectively do things we couldn’t dream of before
  – Uncomfortably fast evolution = revolution
Content Optimization
Content Recommendation on Web Portals

Key Features

Package Ranker (CORE)
Ranks packages by expected CTR based on data collected every 5 minutes

Dashboard (CORE)
Provides real-time insights into performance by package, segment, and property

Mix Management (Property)
Ensures editorial voice is maintained and user gets a variety of content

Package rotation (Property)
Tracks which stories a user has seen and rotates them after user has seen them for a certain period of time

Key Performance Indicators

Lifts in quantitative metrics
Editorial Voice Preserved
### CORE Dashboard: Segment Heat Map

<table>
<thead>
<tr>
<th>Package</th>
<th>male</th>
<th>female</th>
<th>OMG</th>
<th>BUAuto</th>
<th>BUEnt</th>
<th>BU Fin</th>
<th>Health</th>
<th>BUSport</th>
<th>NBA</th>
<th>BUTrav</th>
<th>ALL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Note: The table above represents the Segment Heat Map for various packages. Each cell indicates the percentage or value for that specific parameter. The images are placeholders and do not reflect actual data.*
Telemetry

• **Data:** Time series of logs from user activity and system probes (live and historical archives)
  – CTP (Customer Touch Points) data
  – STP (System Touch Points) data

• **Goal:**
  – Determine possible causes of outages, particularly the long ones
  – Predictive and forensic

• **Planned steps:**
  – Identification of the team that can resolve an outage
  – Visualization of time series to understand long outages that are difficult to resolve
  – Discover and learn patterns associated with outage trends and use them to predict outages

Big Data

So, what should we build?
The Digital Shoebox

SQL / Hive /MR

Stream Processing

Business Intelligence

Machine Learning

Capture any data, react instantaneously, store for later
Use any analysis tool (anywhere, in any combination, interactively)
Collaborate/Share selectively
Building a Digital Shoebox

Relational Query Engine

Relational operators

Machine Learning

Scale-out Compute Fabric

Tiered Storage
Questions

• What is the right balance between common building blocks and custom analytic engines?

• What is the right layering to achieve this?
  – Storage vs. compute tiers; networking trends
  – Scheduling of shared resources
  – Multi-tenanted services
  – Security
  – Evaluation: What is good enough?
Challenges

• Volume
  – Elastic scale-out

• Variety
  – Trade-off: Shared building blocks vs. custom engines
  – Metadata management
    • Many catalogs at many layers (files, tables, docs)
    • Many owners (federation, integration, access control)

• Velocity
  – Real-time and OLTP, interactive, batch
Challenges

• Multi-tenant services
  – HA, rolling upgrades
  – Security (Authentication, isolation, intrusion, DOS)

• CRUD workloads
  – How closely can we couple these with analytics?

• Federated access
  – Bring external data in for analysis
  – Apply analysis in-situ to data elsewhere
Tiered Storage

Dave Campbell, Sriram Rao, XCG
How Far Away is Data?

• GFS and Map-Reduce:
  – Schedule computation “near” data
  – i.e., on machines that have data on their disks
• But
  – Windows Azure Storage
    • And slower tiers such as tape storage ...
  – Main memory growth
    • And flash, SSDs, NVRAM etc. ...
• Must play two games simultaneously:
  – Cache data across tiers, anticipating workloads
  – Schedule compute near cached data
Scale-Out Compute Fabric
YARN and REEF
YARN

• Resource manager for Hadoop2.x
• Allocates compute containers to competing jobs
  – Not necessarily MR jobs!
• Other RMs include Corona, Mesos, Omega
REEF

• Relies on YARN resource manager
  – Can re-target to other RMs

• Evaluator: YARN container with REEF services
  – Capability-awareness, Storage support, Fault-handling support, Communications, Job/task tracking, scheduling hooks

• Activity: User Code to be executed in an Evaluator
  – Monitored, preemptable, re-started as needed
  – Unique id over lifetime of job
  – Executes in an Evaluator, which can be re-used
The Team
MapReduce library

- Runs Hive and Pig
- Excellent starting point for M/R optimizations: Caching, Shuffle, Map-Reduce-Reduce, Sessions, ...

Machine Learning algorithms

- Scalable implementations: Decision Trees, Linear Models, Soon: SVD
- Excellent starting point for: Fault awareness in ML
Popular schedulers

CapacityScheduler

FairScheduler

Deadline-oriented scheduling

New idea:

Support work-preserving preemption

(via) checkpointing ➞ more than preemption
Previous Work
(Amoeba: SoCC’12)

- Amoeba, a lightweight mechanism for enabling elasticity in data-intensive compute frameworks
- Add work-conserving preemption via a “checkpoint/restart” mechanism that saves task output
- Our observation:
  - A reduce task processes a group of keys, one at a time
  - A “key boundary” is a split point—where a reduce task execution can be safely terminated, and a new task can be spawned for the remaining work
- Resource consumption of jobs is elastic
  - Scale up/down usage based on cluster resource availability
  - Preliminary results show that Amoeba can speed up jobs by 33%

- Build on previous work to add preemption to YARN and focus on scheduling
Dynamic Optimization

Leveraging checkpointing for parsimonious scheduling in MR
Killing Tasks vs. Preemption

33% Improvement
Contributing to Apache

Engaging with OSS
- talk with active developers
- show early/partial work
- small patches
- ok to leave things unfinished
Collaborations

• AIP
• GSL
• Isotope team
• Galen Hunt’s team (Drawbridge)
• MSR, XCG