Benchmarking Challenges with Big Data and Cloud Services

Raghu Ramakrishnan
Cloud Information Services Lab (CISL)
Microsoft

The World Has Changed

- Serving applications that need:
  - Scalability!
    - Elastic on demand, commodity boxes
  - Flexible schemas
  - Geographic distribution/replication
  - High availability
  - Low latency
- Are willing to trade:
  - Complex queries
  - ACID transactions
    - But still benefit from support for data consistency

The World Has Changed

- Analytic applications need:
  - Scalability!
    - Elastic on demand, commodity boxes
  - Data variety
  - Wide range of analytics
  - High availability
  - Interactivity
- And are increasingly coupled tightly with data serving and stream capture!
  - Real-time response

Analytics: Hadoop MapReduce Primer

Good for scanning/sequentially writing/appending to huge files

Scales by "mapping" input to partitions, "reducing" partitions in parallel

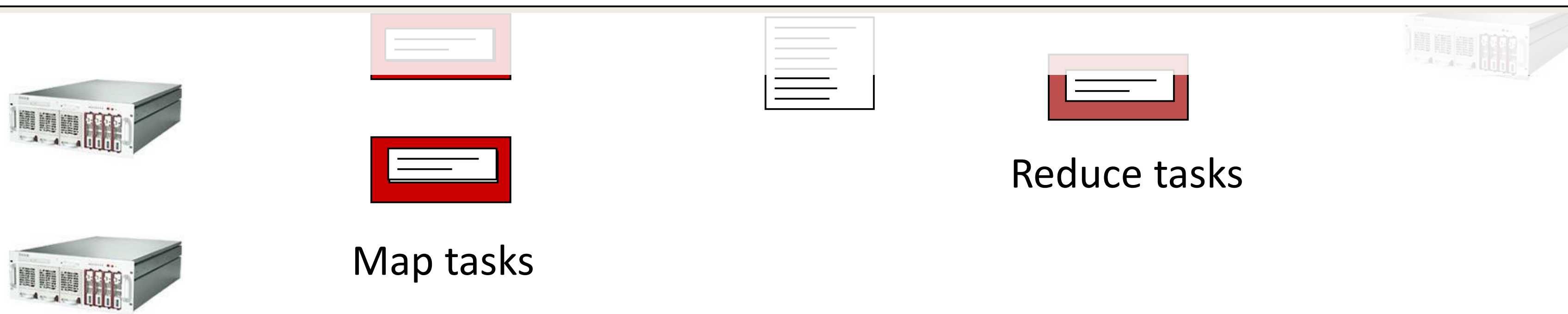
Partitions written to disk for fault-tolerance

Expensive "shuffle" step between Map & Reduce

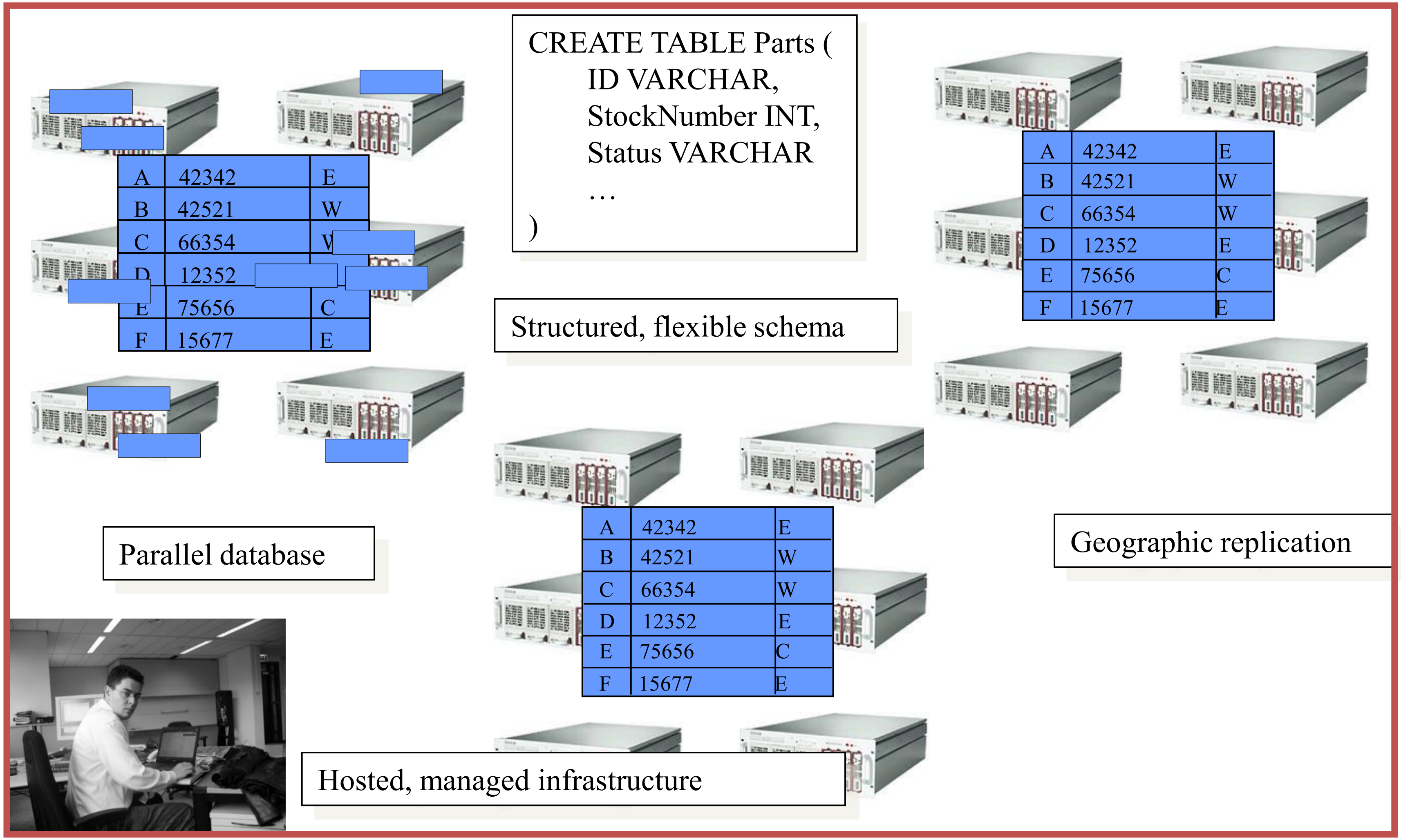
No concept of iteration

Hive and Pig are SQL variants implemented by translation to MapReduce

Not great for serving (reading or writing individual objects)



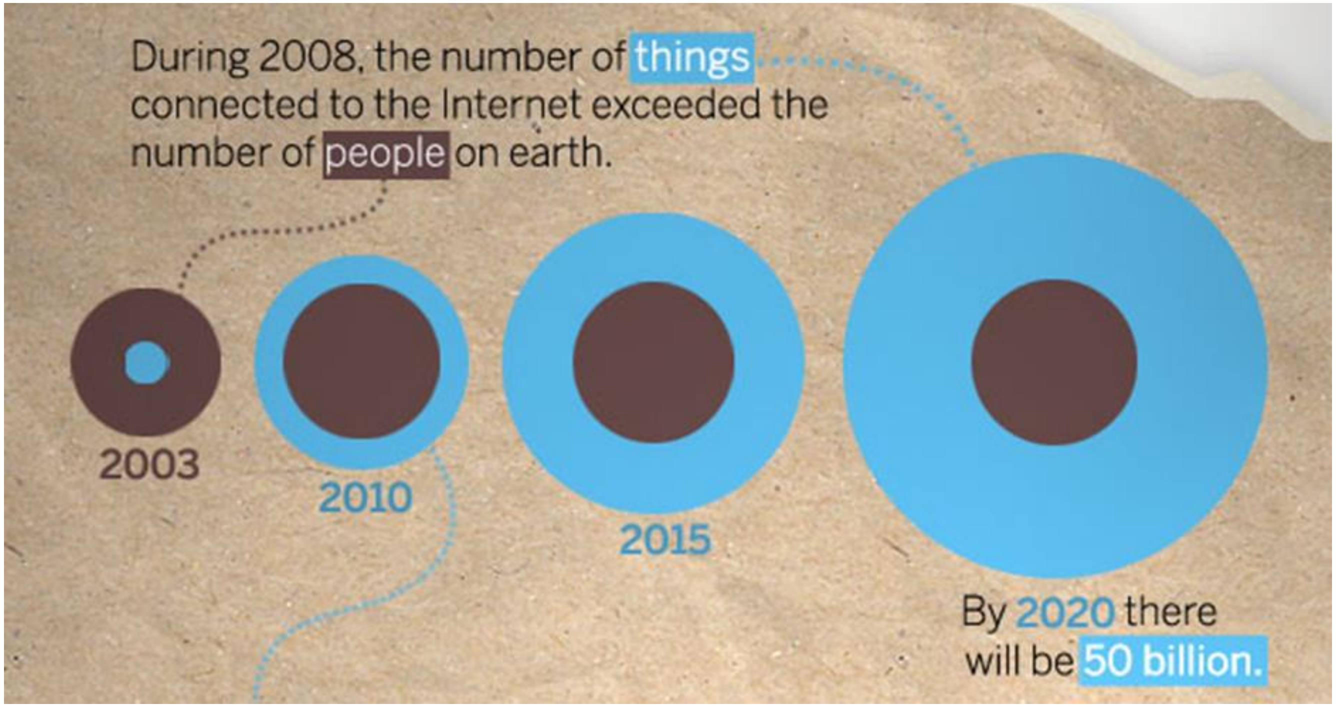
Serving: PNUTS/Sherpa Primer



New Scenarios

Variety, Velocity, Volume

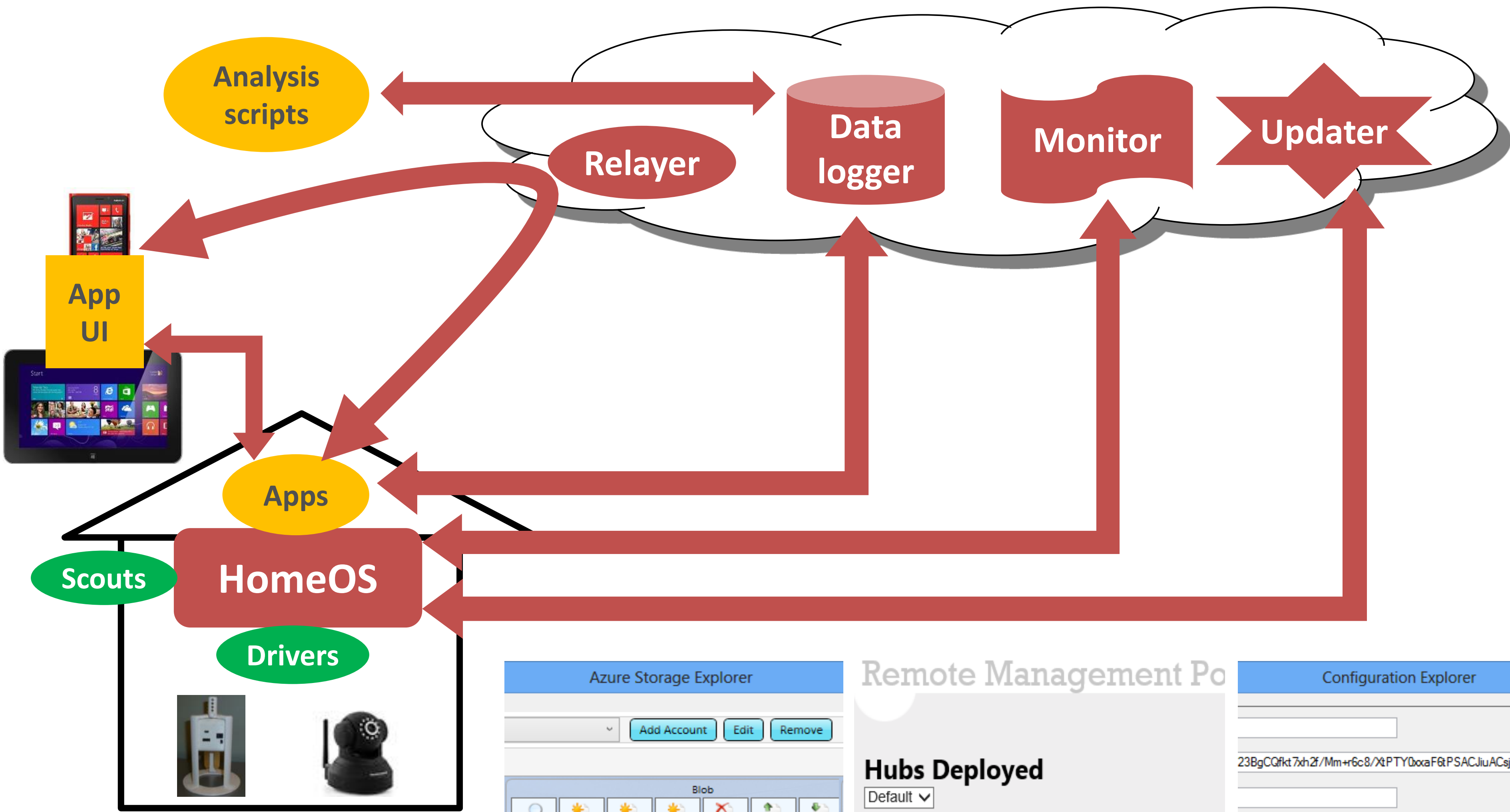
Internet of Things

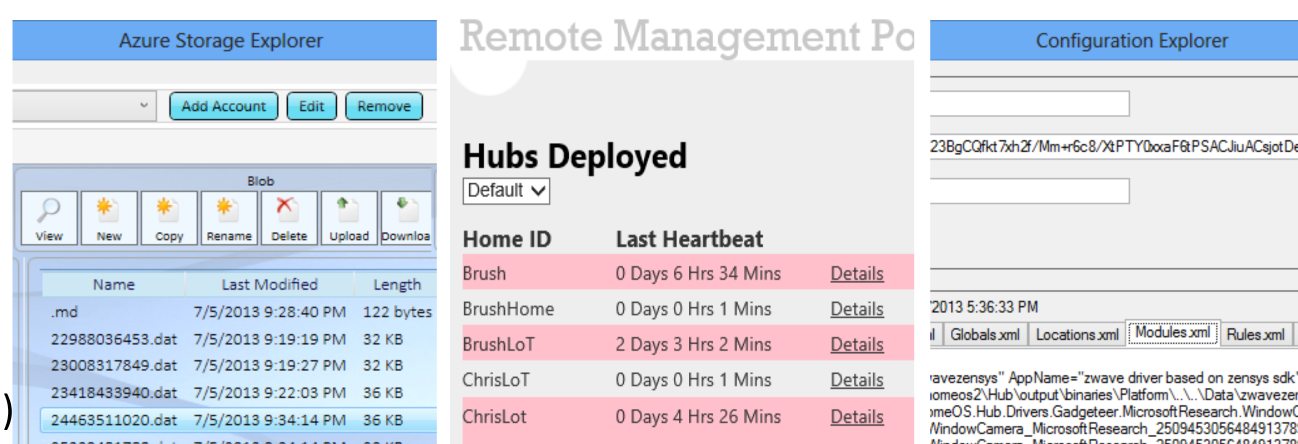


http://blogs.cisco.com/news/the-internet-of-things-infographic/

- IoT opens new "field of streams": new app possibilities
  - Requires real-time responses, continuous forensics
  - Edge processing vs. collection-side processing

HomeOS: An Instance of IoT

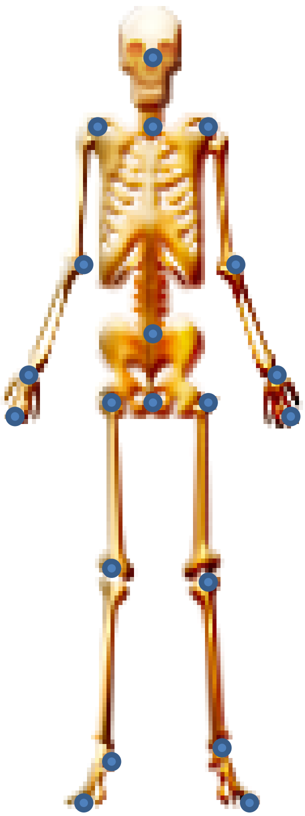




(Slide courtesy Ratul Mahajan, MSR)

Kinect

- The Kinect is an array of sensors.
  - Depth, audio, RGB camera …
- SDK provides a 3D virtual skeleton.
  - 20 points around the body, 30 fps
  - 30 frames per second
  - Between 60-70M sold by May 2013
- Exemplar of "Internet of Things"
  - Event streams from a multitude of devices, enabling broad new apps
    - ML for full-body gait analysis (Mickey Gabel, Ran Gilad-Bachrach, Assaf Schuster, Eng. Med. Bio. 2012)



(Slide modified from Assaf Schuster, Technion)

Typical Y! Applications

- User logins and profiles
  - Including changes that must not be lost!
    - But single-record "transactions" suffice
- Events
  - Alerts (e.g., news, price drops)
  - Social network activity (e.g., user goes offline)
  - Ad clicks, article clicks
- Application-specific data
  - Postings in message board
  - Uploaded photos, tags
  - Shopping carts

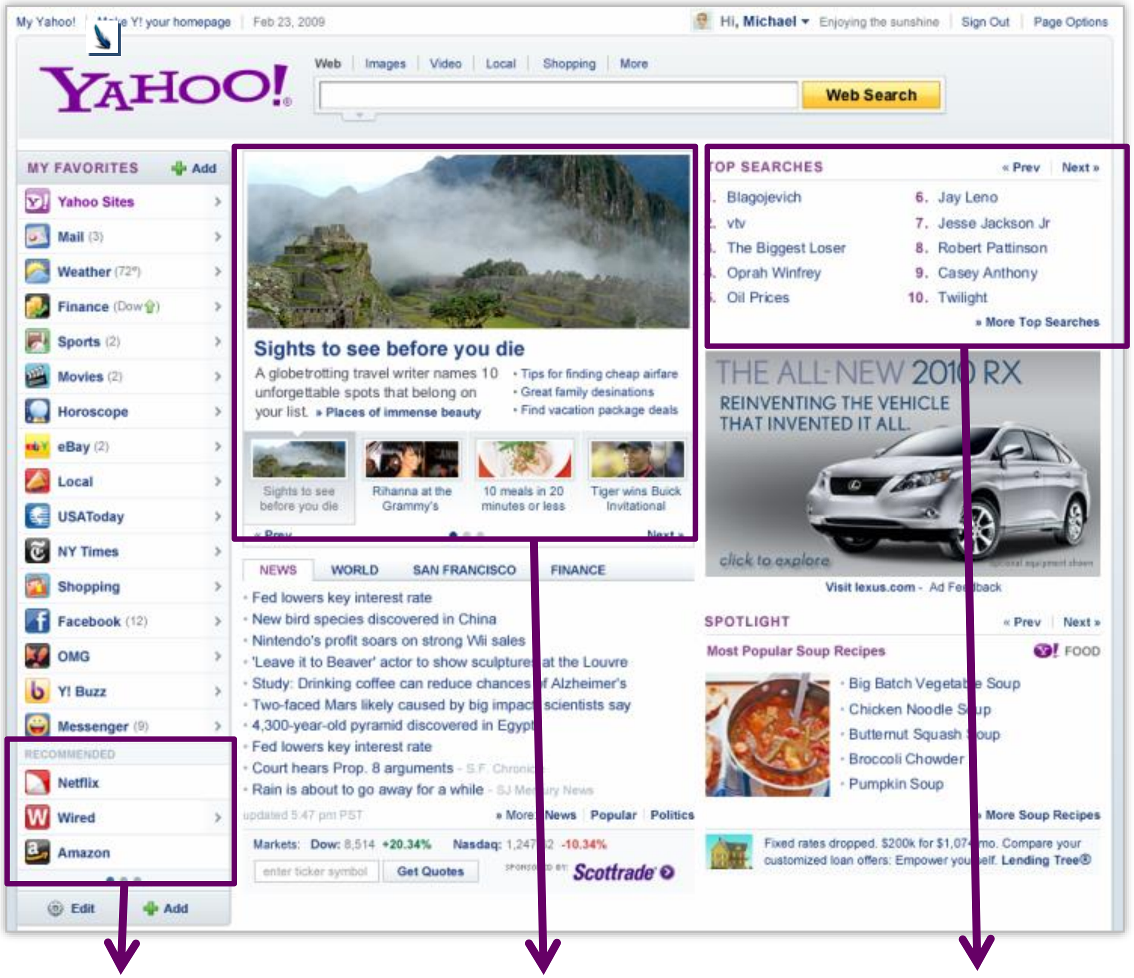
These will be increasingly reflected in enterprise settings as cloud adoption grows, e.g., O365, SalesForce

700M+ UU, 11B pages/month
Hundreds of petabytes of storage
Hundreds of billions of objects
Hundred of thousands of reqs/sec
Global, rapidly evolving workloads

Content Optimization

Agrawal et al., CACM 56(6):92-101 (2013)

Content Recommendation on Web Portals



Recommended links **News Interests** **Top Searches**

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

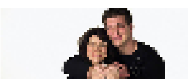
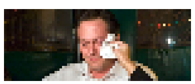
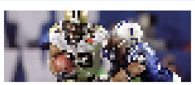
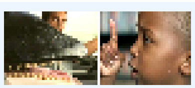
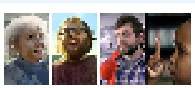
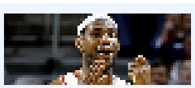
Key Performance Indicators

Lifts in quantitative metrics

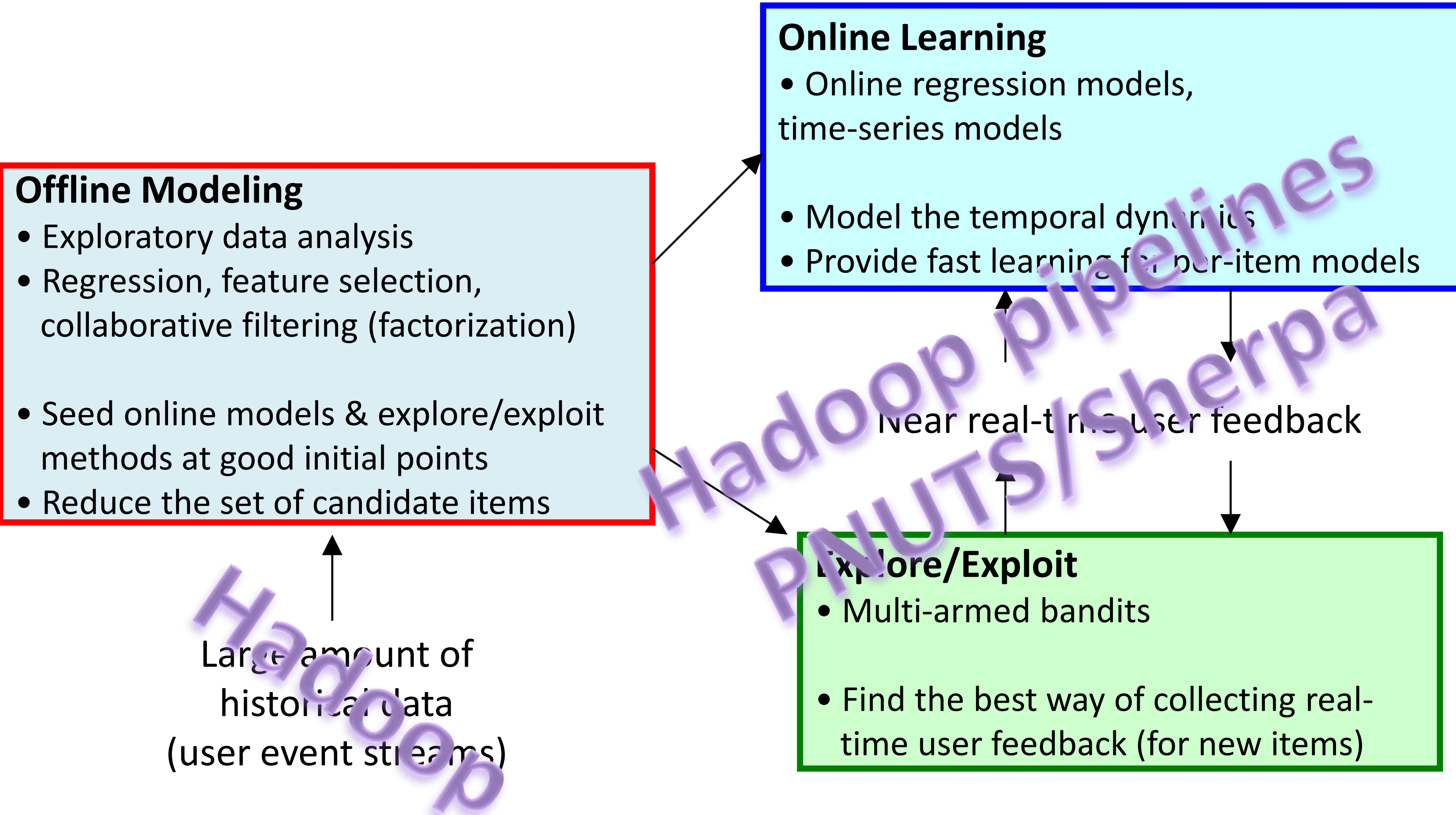
Editorial Voice Preserved

CORE Dashboard

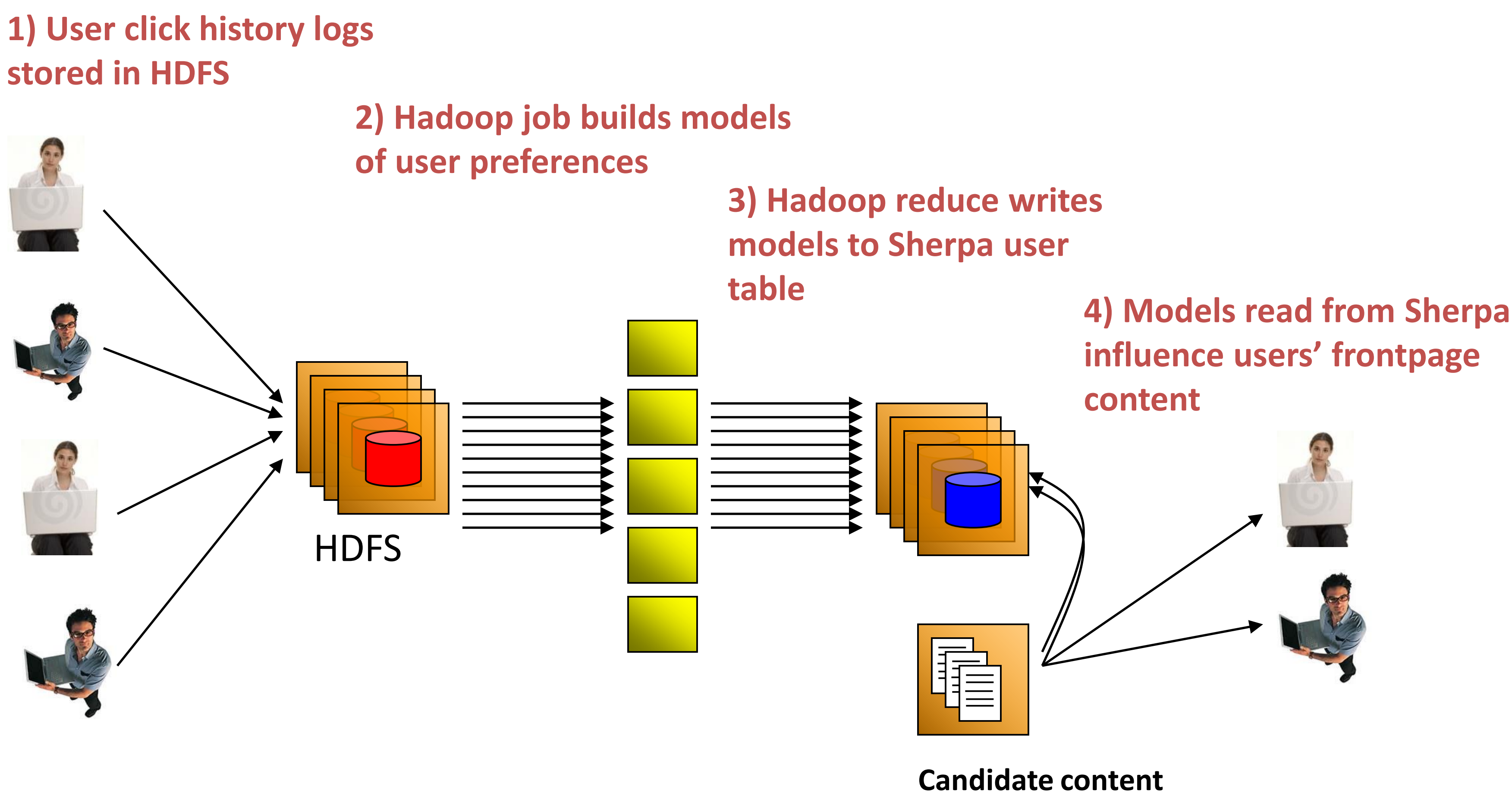
Segment Heat Map

Package	male	female	OMG	BUAuto	BUEnt	BU Fin	Health	BUSport	NBA	BUTrav	ALL
	408,260 18,440 0.0452 8.477	390,404 14,449 0.037 -11.113	270,039 16,940 0.0627 50.661	121,080 7,389 0.061 46.564	270,038 16,940 0.0627 50.661	325,873 20,012 0.0614 47.488	195,796 12,763 0.0652 56.553	350,152 21,454 0.0613 47.152	132,916 9,457 0.0712 70.879	123,388 7,896 0.064 53.691	923,611 38,457 0.0416 0
	1 8,067 852 0.1056 153.654	1 7,657 674 0.088 111.405	1 5,125 720 0.1405 237.406	1 2,382 286 0.1201 188.362	1 5,125 720 0.1405 237.406	1 6,415 858 0.1337 221.221	1 3,769 532 0.1412 239	1 6,750 917 0.1359 226.272	1 2,585 385 0.1489 257.696	1 2,490 330 0.1325 218.294	1 18,137 1,738 0.0958 130.143
	5 9,968 644 0.0646 55.164	3 12,847 777 0.0605 45.256	2 8,569 885 0.1033 148.043	4 3,529 326 0.0924 121.86	2 8,569 885 0.1033 148.043	3 9,744 922 0.0946 127.252	3 6,067 643 0.106 154.537	2 10,187 1,004 0.0986 136.702	5 3,820 420 0.1099 164.058	2 4,037 433 0.1073 157.598	4 25,744 1,595 0.062 48.798
	2 3,326 249 0.0749 79.8	5 3,954 212 0.0536 28.769	5 2,521 231 0.0916 120.066	2 1,004 102 0.1016 143.995	5 2,521 231 0.0916 120.066	5 3,016 276 0.0915 119.782	5 1,860 186 0.1 140.167	3 3,291 310 0.0942 126.229	3 1,141 136 0.1192 186.264	3 1,039 100 0.0962 131.152	3 8,500 541 0.0636 52.859
	11 2,562 133 0.0519 24.677	13 2,004 81 0.0404 -2.926	3 1,250 122 0.0976 134.403	6 629 51 0.0811 94.73	3 1,250 122 0.0976 134.403	4 1,608 151 0.0939 125.53	2 919 103 0.1121 169.175	4 1,669 154 0.0923 121.604	4 655 74 0.113 171.334	4 591 55 0.0931 123.506	10 5,342 252 0.0472 13.295
	3 2,881 206 0.0715 71.727	2 3,242 230 0.0709 70.384	4 2,071 196 0.0946 127.295	3 949 95 0.1001 140.42	4 2,071 196 0.0946 127.295	2 2,614 254 0.0972 133.368	4 1,605 165 0.1028 146.901	5 2,740 239 0.0872 109.489	10 1,036 94 0.0907 117.912	9 958 78 0.0814 95.543	2 7,043 493 0.07 68.114
	6 10,785 649 0.0602 44.523	4 12,768 742 0.0581 39.571	7 8,580 694 0.0809 94.261	7 3,511 283 0.0806 93.584	7 8,580 694 0.0809 94.261	6 9,725 795 0.0817 96.332	6 6,138 550 0.0896 115.204	6 10,670 866 0.0812 94.925	11 3,669 321 0.0875 110.122	5 3,785 339 0.0896 115.104	5 27,331 1,641 0.06 44.2
	10 22,202 1,212 0.0546 31.106	7 23,328 1,200 0.0514 23.543	6 15,593 1,289 0.0827 98.535	5 6,552 533 0.0813 95.374	6 15,593 1,289 0.0827 98.535	7 17,652 1,376 0.078 87.214	8 10,797 915 0.0847 103.532	7 19,050 1,522 0.0799 91.882	9 6,639 604 0.091 118.498	7 6,435 552 0.0858 106.018	6 52,978 2,786 0.0526 26.299
	22 26,685 1,160 0.0435 4.401	10 35,405 1,530 0.0432 3.786	8 19,832 1,572 0.0793 90.371	9 7,844 552 0.0704 69.011	8 19,832 1,572 0.0793 90.371	8 21,743 1,641 0.0755 81.26	7 13,721 1,167 0.0851 104.267	8 22,168 1,743 0.0786 88.836	8 8,249 788 0.0955 129.424	8 8,327 689 0.0827 98.721	18 74,559 3,167 0.0425 2.014
	4 7,745 518 0.0669 60.628	26 7,202 185 0.0257 -38.308	13 4,898 322 0.0657 57.889	15 2,308 148 0.0641 54.007	13 4,898 322 0.0657 57.889	11 6,051 423 0.0699 67.891	19 3,652 235 0.0643 54.544	9 6,436 506 0.0786 88.82	2 2,562 308 0.1202 188.726	12 2,359 169 0.0716 72.057	7 17,235 834 0.0484 16.217
	7 7,699 460 0.0597 43.495	29 7,201 169 0.0235 -43.635	11 4,809 340 0.0707 69.8	10 2,269 158 0.0696 67.239	11 4,809 340 0.0707 69.8	9 6,004 433 0.0721 73.205	14 3,544 243 0.0686 64.674	10 6,247 475 0.076 82.615	6 2,482 257 0.1035 148.682	11 2,329 167 0.0717 72.211	12 17,169 783 0.0456 9.529
	12 7,688 393	9 7,229 336	9 4,785 363	17 2,280 139	9 4,785 363	12 6,037 403	12 3,501 245	11 6,319 430	15 2,397 182	15 2,312 152	8 17,275 833

CORE Modeling Overview



Data Management in CORE



CORE Data Management

- Table of user profiles
- Read: When determining what story to show
- Write: After user action
- Write: After grid computation

Sherpa

User	Profile
Adam	41,311,56,12,13
Brad	42,15,66,123,1
Toby	4321,1,44,13
Utkarsh	42,133,122,33
…	…

Example: User Activity Modeling

Input: Large dimensionality vector describing possible user activities

- But a typical user has a sparse activity vector

Output: User profile that weights affinity along dimensions/activities of interest

Pipeline steps:

- Example formation:
  - Data acquisition and sessionization
  - Feature and target generation
- Model training
- Model testing
- Deployment: Upload models for serving

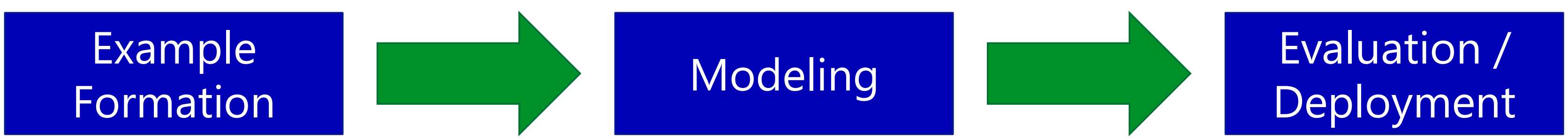
Machine Learning Workflow

Step I: Example Formation

- Feature Extraction
- Label Extraction

Step II: Modeling

Step III: Deployment (or just Evaluation)

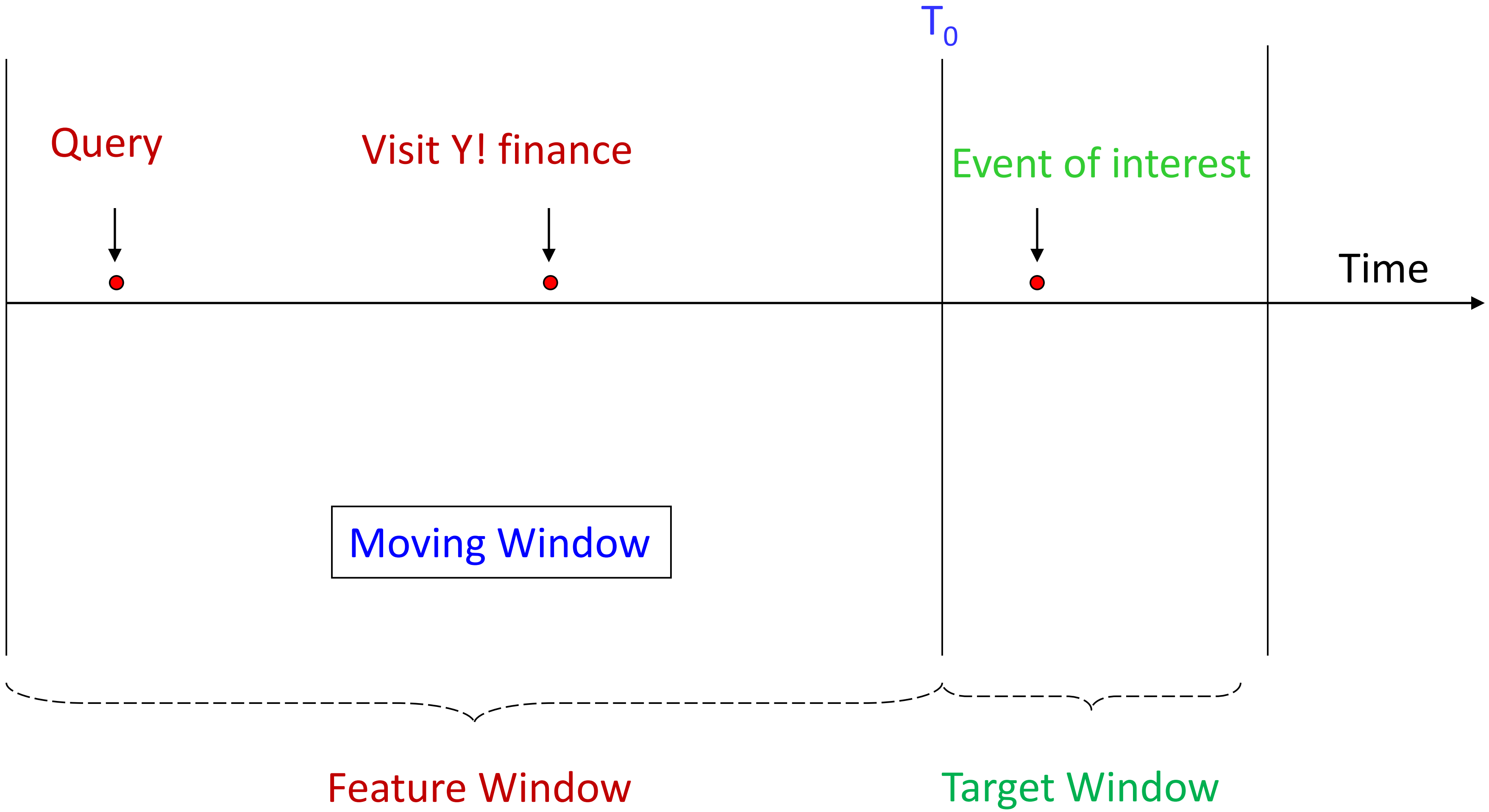


User Activity Modeling

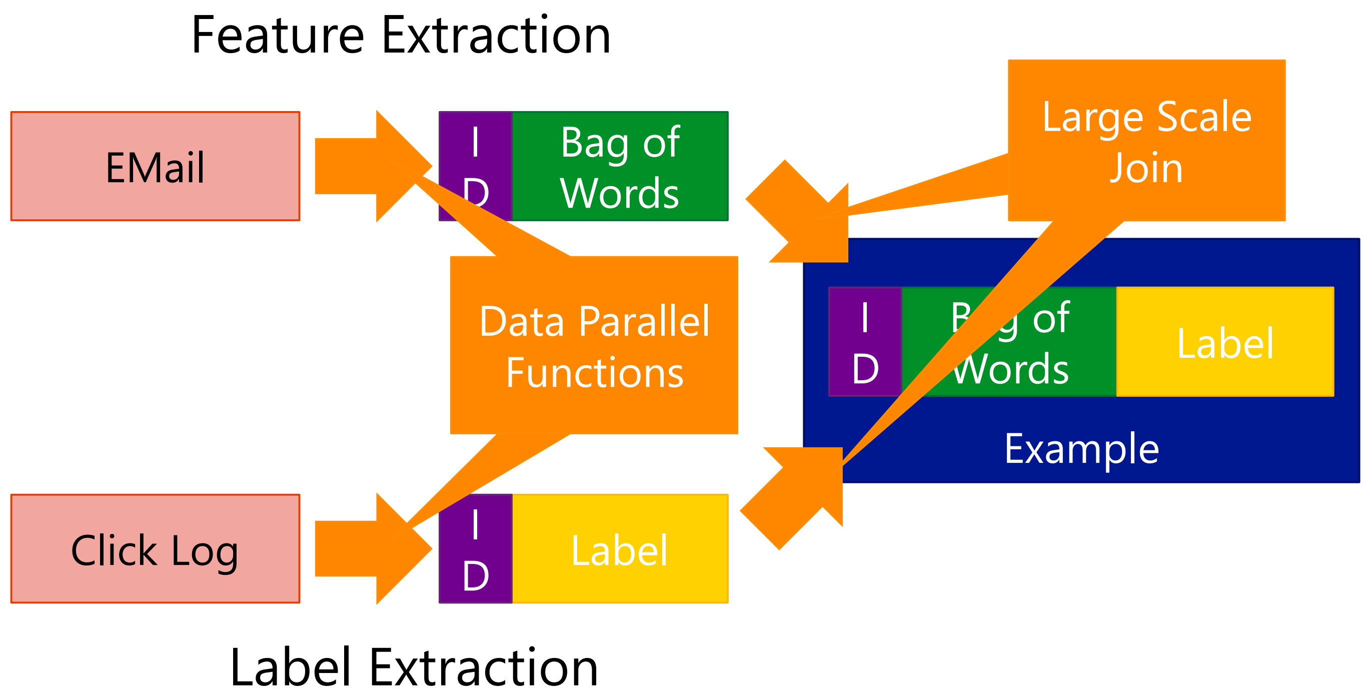
Attribute	Possible Values	Typical values per user
Pages	~ MM	10 – 100
Queries	~ 100s of MM	Few
Ads	~ 100s of thousands	10s

- Hadoop pipeline to model user interests from activities
- Basis for Deep Analysis Pipeline proposal for Big Data benchmark from Bhandarkar (based on collaboration with Vijay Narayanan)

Feature and Target Windows



Example Formation: SQL at Scale



Given that click/target rates are very low (0.01 to 1%), good idea to filter out email from time windows with no clicks <u>before</u> doing the join

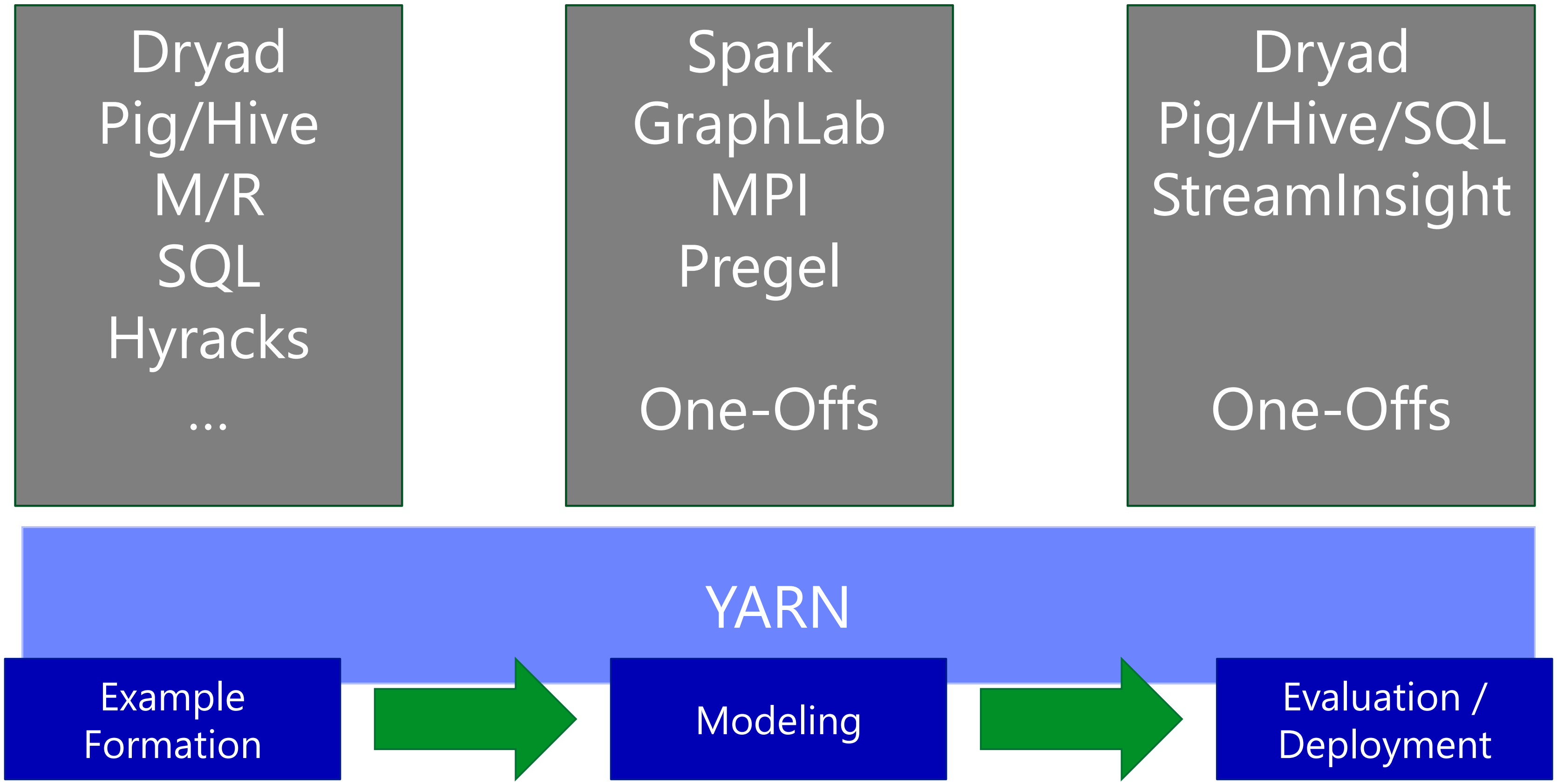
User Modeling Pipeline

Component	Data Processed	Time
Data Acquisition	~ 1 Tb per time period	2 – 3 hours
Feature and Target Generation	~ 1 Tb * Size of feature window	4 - 6 hours
Model Training	~ 50 - 100 Gb	1 – 2 hours for 100's of models
Scoring	~ 500 Gb	1 hour

Model Training

- Once examples have been formed, can use any available techniques to train models:
  - Gradient Boosted Decision Trees
  - Naïve Bayes
  - Linear Regression
  - SVMs
- Models are cross-validated to find good ones
- Finally, models are operationalized by deploying to serving systems

Machine Learning Workflow

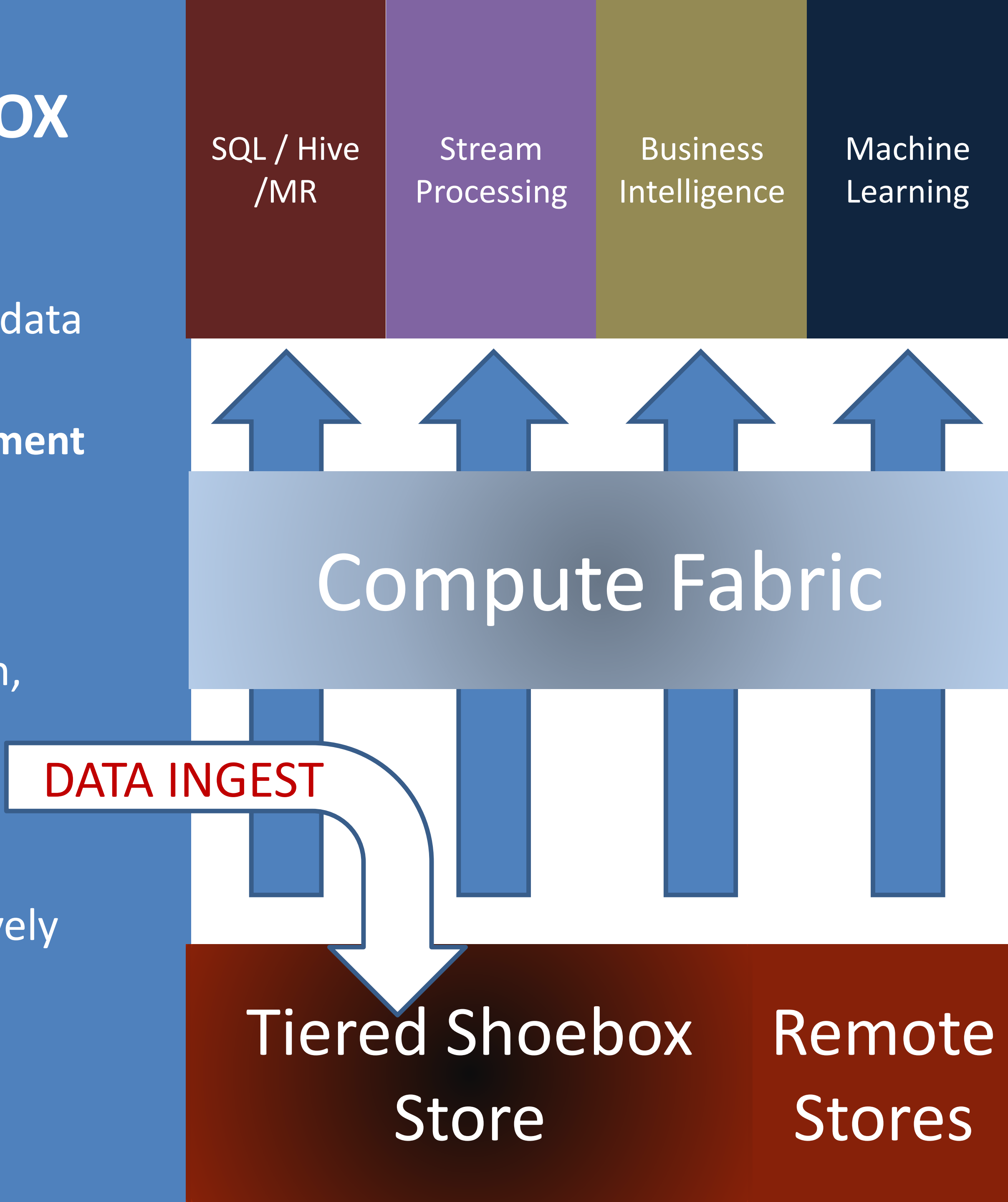


The Digital Shoebox

Build it—they're here already!

THE DIGITAL SHOEBOX

- Capture any data, react instantaneously, mix with data stored anywhere
  - **Tiered storage management**
  - **Federated access**
- Use any analysis tool (anywhere, mix and match, interactively)
  - **Compute fabric**
- Collaborate/Share selectively



MICROSOFT

POLYBASE

SQL Over Relational Tables and Hadoop

POWER BI

Interactive Discovery and Exploration

HDINSIGHT

Hadoop on Azure

Integrated Query "In-Place"

Can join and group-by tables from a relational source with tables in a Hadoop cluster without needing to learn MapReduce

Integrated BI Tools

Using Excel, end users can search for data sources with Power Query and do roll-up/drill-down etc. with Power Pivot—across both relational and Hadoop data

Interactive Visualizations

Use Power View for immersive interactivity and visualizations of both relational and Hadoop data

A COMMON VISION

The vision of supporting many kinds of scalable analytics over all of a user's data is shared by many vendors

Aster/Teradata

Berkeley Data Analytics Stack

Cloudera

Google

HortonWorks

Microsoft

Pivotal/EMC

SQL on Hadoop panel, Aug 2013:
http://hivedata.com/real-time-query-panel-discussion/

Challenges

- Volume
  - Elastic scale-out
  - Multi-tenancy
- Variety
  - Data variety coupled with range of analytics
- Velocity
  - Real-time and OLTP, interactive, batch

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, e.g., Glacier …
  - Main memory growth
    - And flash, SSDs, NVRAM etc. …
- Must play two games simultaneously:
  - Cache data across tiers, anticipating workloads
  - Schedule compute near cached data

Compute Fabric: YARN

- Resource manager for Hadoop2.x
- Allocates compute containers to competing jobs
  - Not necessarily MR jobs!
  - Containers are the unit of resource
  - Can fail or be taken away; programmer must handle these cases
- Other RMs include Corona, Mesos, Omega

Making YARN Easier to Use: REEF

- 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

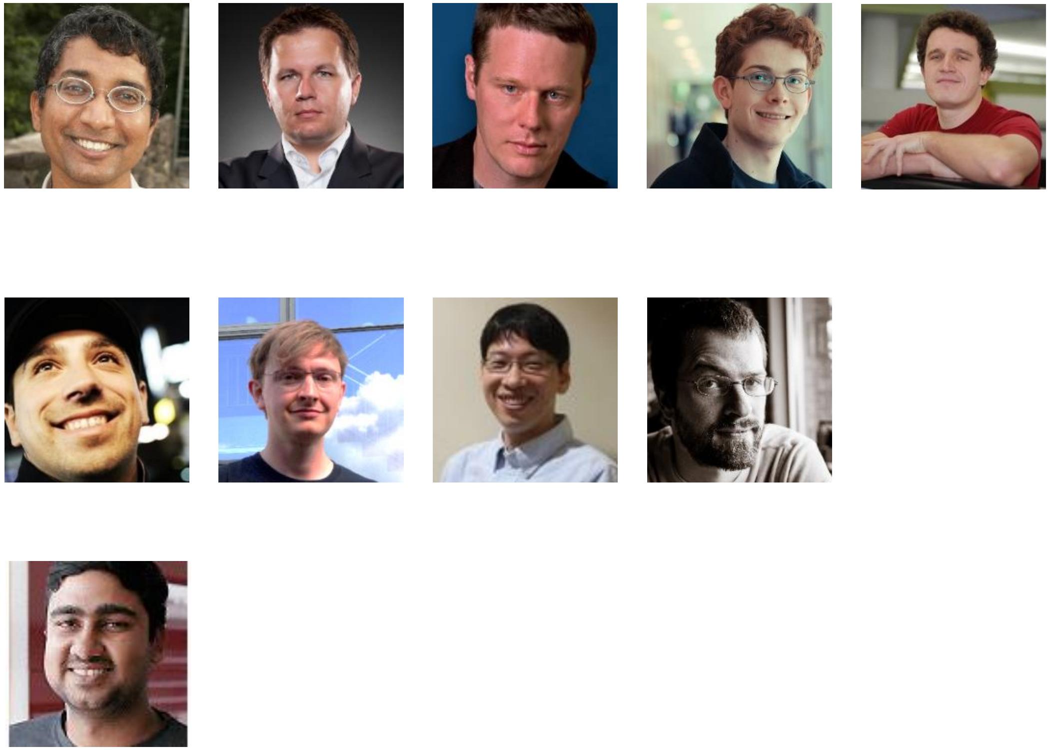
REEF

Retainable

Evaluator

Execution

Framework

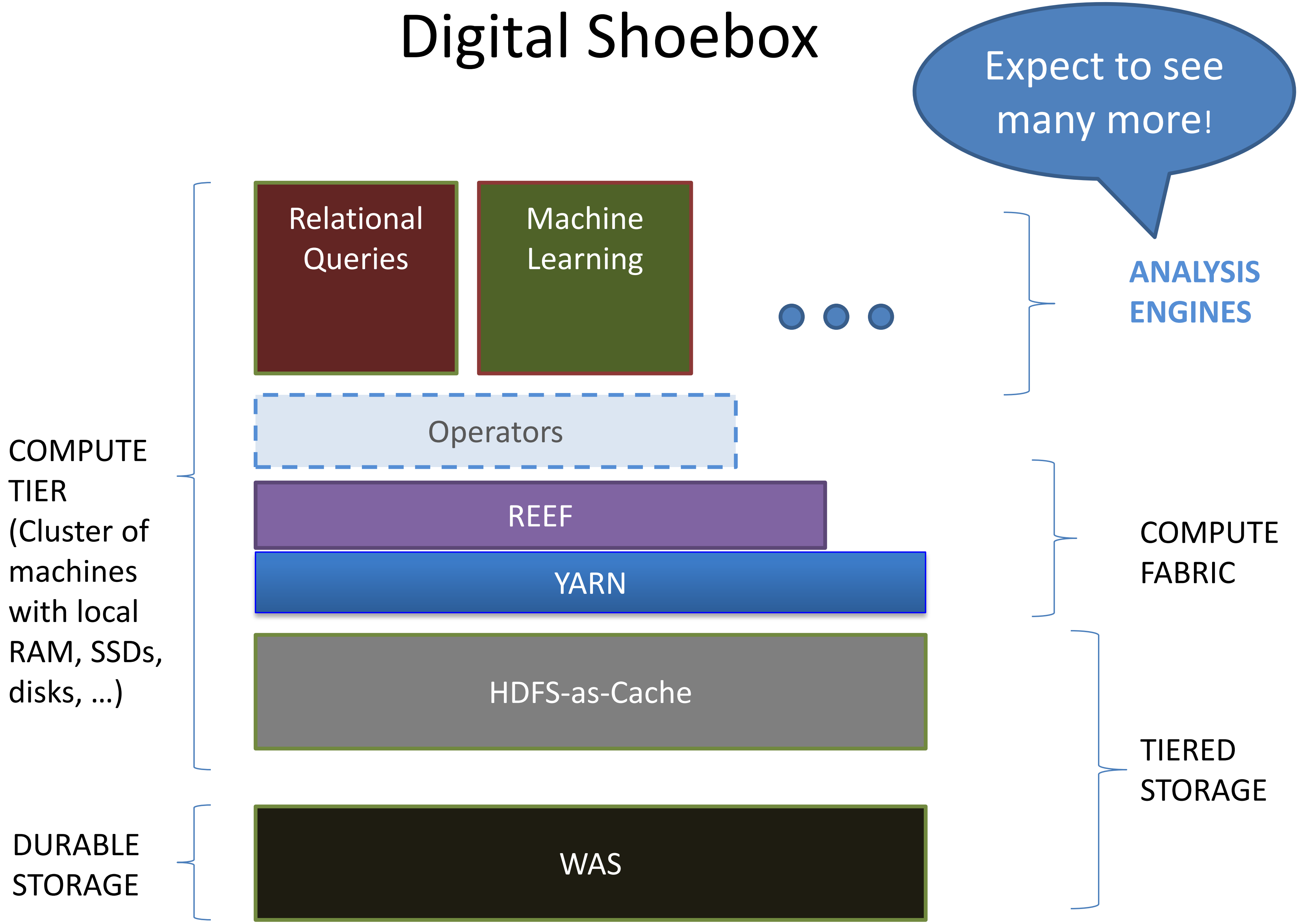


Open-source release planned

The Team @ Microsoft

Demo at VLDB

Digital Shoebox



Benchmarking Big Data

Clouds, Quality, Variety, Velocity

Building on TPC, TREC, SPEC
Recent initiatives: WBBD, BigDataTop100
This workshop!

Benchmark Dimensions

- Workload dimensions
  - Data variety (Tables, graphs, streams, loosely-structured docs, media)
  - Type of analysis (serving vs. analytics; degree of consistency; quality-sensitivity; batch vs. interactive vs. real-time)
  - Result quality vs. performance
- System dimensions
  - Architecture (Storage hierarchy, edge processing)
  - Cloud (Elasticity)
- Metrics
  - Performance (latency/throughput, stream rate)
  - Scale-up, scale-out, elasticity
  - Quality (precision-recall, ranking quality, lift)
  - Availability (uptime, range of faults handled, fault-recovery time)
  - Cost: $, $/perf metric, per metric/$

YCSB: Benchmarking Serving Systems

citation

- There are many “cloud DB” and “nosql” systems out there
  - Sherpa
  - BigTable
    - HBase, Hypertable, HTable
  - Megastore
  - Azure
  - Cassandra
  - Amazon Web Services
    - S3, SimpleDB, EBS
  - CouchDB
  - Voldemort
  - Dynomite
  - Espresso
- How do they compare?
  - Feature tradeoffs
  - Performance tradeoffs
  - Not clear!

Goal

- Implement a standard benchmark for data serving
  - Evaluate different systems on common workloads
  - Focus on performance and elastic scale out
    - Future additions – availability, replication
    - Not to mention multi-tenancy and “services”!
- Artifacts
  - Open source workload generator
  - Experimental study comparing several systems

Benchmark Tiers

- Tier 1 – Performance
  - For constant hardware, increase offered throughput until saturation
  - Measure resulting latency/throughput curve
  - “Sizeup” in Wisconsin benchmark terminology
- Tier 2 – Scalability
  - Scaleup – Increase hardware, data size and workload proportionally. Measure latency; should be constant
  - Elastic speedup – Run workload against N servers; while workload is running add N+1th server; measure timeseries of latencies (should drop after adding server)

Workloads

- Workload – particular combination of workload parameters, defining one workload
  - Defines read/write mix, request distribution, record size, …
  - Two ways to define workloads:
    - Adjust parameters to an existing workload (via properties file)
    - Define a new kind of workload (by writing Java code)

- Experiment – running a particular workload on a particular hardware setup to produce a single graph for 1 or N systems
  - Example – vary throughput and measure latency while running a workload against Cassandra and HBase

- Workload package – A collection of related workloads
  - E.g., CoreWorkload – a set of basic read/write workloads

Tier 1 CoreWorkload

- CoreWorkload defines:
  - A parameterized data set
  - A parameterized query
    - Roughly: do a read, write, insert or scan with some probability on each request
  - A set of parameters for the data set and queries
  - This is sufficient to run a wide range of specific Workload instances
    - E.g., 95/5 read/write, 95/2.5/2.5 read/write/insert, etc
- What if I want something other than these workloads?
  - Abstract Workload class can be extended in YCSB with your own data set and query by writing Java code

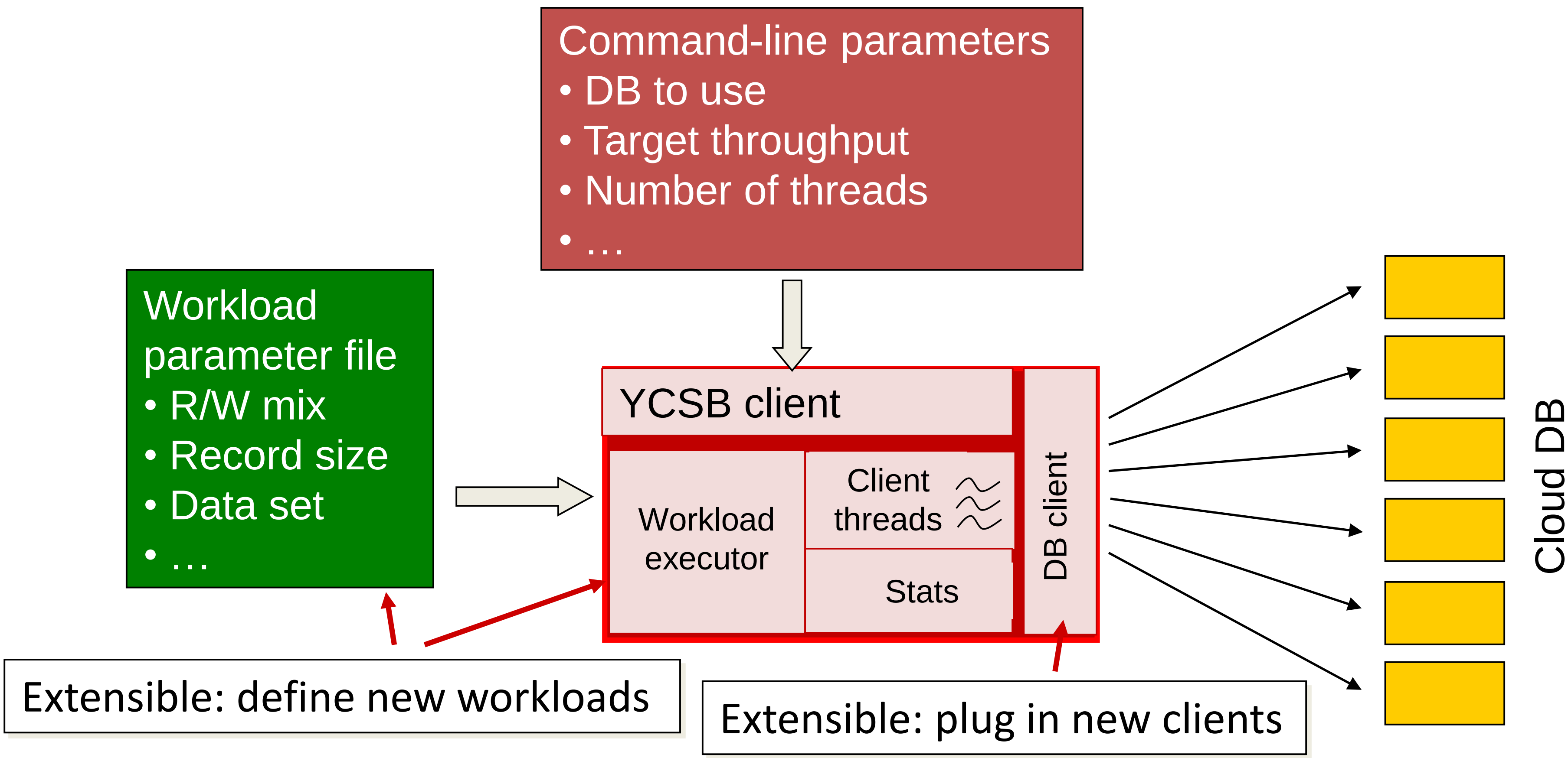
Core Workload Package

Goal: Define handful of workloads as the core "standard" workloads

- Workload A – Update heavy
  - 50/50 read/write
  - Update part of the record
  - Zipfian request distribution
  - Example app: session store recording recent actions
- Workload B – Read mostly
  - 95/5 read/write
  - Update whole record
  - Zipfian request distribution
  - Example app: photo tagging; add a tag is an update, but most operations are to read tags
- Workload C – Read only
  - 100% read
  - Zipfian request distribution
  - Example app: user profile cache, where profiles are constructed elsewhere (e.g., Hadoop)
- Workload D - Read latest
  - 95/0/5 read/write/insert
  - "Latest" request distribution
  - Example app: Twitter event store
- Workload E – Short ranges
  - 95/5 scan/insert
  - Zipfian request distribution
  - Example app: threaded conversations, where each scan is for the posts in a given thread (assumed to be clustered by thread id)
  - Note – inserts should be random LoadOrder

Benchmark Tool

- Java application
  - Many systems have Java APIs
  - Other systems via HTTP/REST, JNI or some other solution



GridMix: Benchmarking Hadoop Analytics

citation

- Mix of synthetic jobs modeling a profile mined from production loads
- Emulates users and job queues
- Can emulate distributed cache files
- Can emulate (de-)compression, high-RAM jobs, resource usage
- Simplifying assumptions about:
  - File-system properties (other than bytes/records consumed/emitted)
    - Record sizes / key distributions based on averages, i.e., no skew
  - Job I/O rates and memory profiles
  - Jobs assumed to succeed; run independently of other jobs

TEXTURE: Benchmarking Performance of Text Queries on a Relational DBMS

Ercegovac, DeWitt, Ramakrishnan SIGMOD 05

- Queries with relevance ranking, instead of those that compute all answers
  - Richer mix of text and relational processing
  - Measures only performance, not quality
  - Only queries; no updates, bulk-loading, or multi-user support
- Micro-benchmark where experiment is defined by selecting:
  - Dataset size: Data schema based on Wisconsin Benchmark, extending it with two (short, in-line with row; long, separate blob) text fields generated using TextGen
  - Query workload: (1) text-only queries, (2) single-table mixed queries, and (3) multiple-table mixed queries.
  - Evaluation mode: (1) all results, (2) the first result, or (3) top-k results

TextGen: Synthetic Text Generator

Ercegovac, DeWitt, Ramakrishnan SIGMOD 05

- Generates large text corpora that reflect (performance related) characteristics of a given "seed" corpus
- Features from seed that are maintained during scale up:
  - Word Distribution W(w,c): Associates with every unique word w in the corpus, the number of times c it appears in the corpus.
    - Modeled by using same proportions as in seed
  - Vocabulary Growth (G): Number of unique words grows as new documents are added to a corpus.
    - Modeled using Heap's law: $G(x) = \alpha x^{\beta}$; parameters estimated using least squares fit
  - Unique Words per Document (U) and Document Length (D)
    - Modeled using averages from seed corpus

BigBench: Benchmarking Hadoop Analytics

Ghazal et al., SIGMOD 13

- End-to-end big data benchmark proposal
- Data schemas extend TPC-DS
  - Semi-structured component: Web clicks
  - Unstructured: Product reviews
- Synthetic data generator
  - Suggestion: Consider TextGen (from Texture!) for unstructured data
- Technical considerations in choosing workload:
  - Data types involved; declarative or procedural; Statistical/mining/SQL
- Analytic workload based on McKinsey retail analytics report
  - Associations, e.g., Cross-selling based on products purchased together
  - Statistical, e.g., correlation of sales with competitor's prices
  - ML, e.g., sentiment analysis of product reviews
  - SQL-based reports, e.g., 30-day sales before and after price change

DAP: Benchmarking ML Pipelines

Milind Bhandarkar with Vijay Narayanan

- Based on user-modeling pipeline workloads at Yahoo!
- Proposal:
  - Pipelines constructed by mix and match of various stages
  - Different analysis/modeling techniques per stage
  - (Create a standardized version and) publish performance numbers for every stage

CONCLUSIONS

Data is the new gold, data mining the new Klondike

Big Data platforms fuse scale-out analytics and serving systems

Moving to the cloud: ComScore for DB services?

Convergence of analytics

- Batch, interactive, real-time

Digital Shoebox trend

- Data variety: Structured, unstructured, streams, graphs, DNA, media, etc.
- Analytics variety: SQL, ML, BI

New things to measure

- Quality
- Elasticity
- Multitenancy