Analyzing Big Data at Twitter

Web 2.0 Expo, 2010
Kevin Weil   @kevinweil
Three Challenges

- Collecting Data
- Large-Scale Storage and Analysis
- Rapid Learning over Big Data
My Background

- Studied Mathematics and Physics at Harvard, Physics at Stanford
- **Tropos Networks** (city-wide wireless): GBs of data
- **Cooliris** (web media): Hadoop for analytics, TBs of data
- **Twitter**: Hadoop, Pig, machine learning, visualization, social graph analysis, PBs of data
Three Challenges

 › Collecting Data
 › Large-Scale Storage and Analysis
 › Rapid Learning over Big Data
Data, Data Everywhere

› You guys generate a lot of data
› Anybody want to guess?
Data, Data Everywhere

- You guys generate a lot of data
- Anybody want to guess?
- 12 TB/day (4+ PB/yr)
Data, Data Everywhere

- You guys generate a lot of data
- Anybody want to guess?
  - 12 TB/day (4+ PB/yr)
  - 20,000 CDs
Data, Data Everywhere

- You guys generate a lot of data
- Anybody want to guess?
  - 12 TB/day (4+ PB/yr)
  - 20,000 CDs
  - 10 million floppy disks
Data, Data Everywhere

- You guys generate a lot of data
- Anybody want to guess?
- 12 TB/day (4+ PB/yr)
- 20,000 CDs
- 10 million floppy disks
- 450 GB while I give this talk
Syslog?

- Started with syslog-ng
- As our volume grew, it didn’t scale
Syslog?

- Started with syslog-ng
- As our volume grew, it didn’t scale
- Resources overwhelmed
- Lost data
Scribe

- Surprise! FB had same problem, built and open-sourced Scribe
- Log collection framework over Thrift
- You “scribe” log lines, with categories
- It does the rest
Scribe

- Runs locally; reliable in network outage
Scribe

- Runs locally; reliable in network outage

- Nodes only know downstream writer; hierarchical, scalable
Scribe

- Runs locally; reliable in network outage
- Nodes only know downstream writer; hierarchical, scalable
- Pluggable outputs
Scribe at Twitter

- Solved our problem, opened new vistas
- Currently 40 different categories logged from javascript, Ruby, Scala, Java, etc
- We improved logging, monitoring, behavior during failure conditions, writes to HDFS, etc
- Continuing to work with FB to make it better

http://github.com/traviscrawford/scribe
Three Challenges

‣ Collecting Data
‣ **Large-Scale Storage and Analysis**
‣ Rapid Learning over Big Data
How Do You Store 12 TB/day?

› Single machine?
› What’s hard drive write speed?
How Do You Store 12 TB/day?

› Single machine?
› What’s hard drive write speed?
› ~80 MB/s
How Do You Store 12 TB/day?

- Single machine?
- What’s hard drive write speed?
- ~80 MB/s
- 42 hours to write 12 TB
How Do You Store 12 TB/day?

› Single machine?
› What’s hard drive write speed?
› ~80 MB/s
› 42 hours to write 12 TB
› Uh oh.
Where Do I Put 12TB/day?

- Need a cluster of machines
- ... which adds new layers of complexity
Hadoop

- Distributed file system
- Automatic replication
- Fault tolerance
- Transparently read/write across multiple machines
Hadoop

- Distributed file system
- Automatic replication
- Fault tolerance
- Transparently read/write across multiple machines
- MapReduce-based parallel computation
- Key-value based computation interface allows for wide applicability
- Fault tolerance, again
Hadoop

- **Open source**: top-level Apache project
- **Scalable**: Y! has a 4000 node cluster
- **Powerful**: sorted 1TB random integers in 62 seconds

- **Easy packaging/install**: free Cloudera RPMs
MapReduce Workflow

- **Challenge:** how many tweets per user, given tweets table?
- **Input:** key=row, value=tweet info
- **Map:** output key=user_id, value=1
- **Shuffle:** sort by user_id
- **Reduce:** for each user_id, sum
- **Output:** user_id, tweet count
- With 2x machines, runs 2x faster
Challenge: how many tweets per user, given tweets table?

Input: key=row, value=tweet info

Map: output key=user_id, value=1

Shuffle: sort by user_id

Reduce: for each user_id, sum

Output: user_id, tweet count

With 2x machines, runs 2x faster
MapReduce Workflow

- Challenge: how many tweets per user, given tweets table?
- Input: key=row, value=tweet info
- **Map:** output key=user_id, value=1
- **Shuffle:** sort by user_id
- **Reduce:** for each user_id, sum
- **Output:** user_id, tweet count
- With 2x machines, runs 2x faster
MapReduce Workflow

- Challenge: how many tweets per user, given tweets table?
- Input: key=row, value=tweet info
- Map: output key=user_id, value=1
- Shuffle: sort by user_id
- Reduce: for each user_id, sum
- Output: user_id, tweet count
- With 2x machines, runs 2x faster
MapReduce Workflow

- Challenge: how many tweets per user, given tweets table?
- Input: key=row, value=tweet info
- Map: output key=user_id, value=1
- Shuffle: sort by user_id
- Reduce: for each user_id, sum
- Output: user_id, tweet count
- With 2x machines, runs 2x faster
MapReduce Workflow

- Challenge: how many tweets per user, given tweets table?
- Input: key=row, value=tweet info
- Map: output key=user_id, value=1
- Shuffle: sort by user_id
- Reduce: for each user_id, sum
- **Output**: user_id, tweet count
- With 2x machines, runs 2x faster
MapReduce Workflow

- **Challenge:** how many tweets per user, given tweets table?
- **Input:** key=row, value=tweet info
- **Map:** output key=user_id, value=1
- **Shuffle:** sort by user_id
- **Reduce:** for each user_id, sum
- **Output:** user_id, tweet count
- **With 2x machines, runs 2x faster**
Two Analysis Challenges

- Compute mutual followings in Twitter's interest graph
  - grep, awk? No way.
  - If data is in MySQL... self join on an n-billion row table?
  - $n,000,000,000 \times n,000,000,000 = ?$
Two Analysis Challenges

- Compute mutual followings in Twitter’s interest graph
- grep, awk? No way.
- If data is in MySQL... self join on an n-billion row table?
- \(n,000,000,000 \times n,000,000,000 = ?\)
- I don’t know either.
Two Analysis Challenges

- Large-scale grouping and counting
  
  ```sql
  select count(*) from users? maybe.
  ```

  ```sql
  select count(*) from tweets? uh...
  ```

  Imagine joining these two.

  And grouping.

  And sorting.

HALP
Back to Hadoop

- Didn’t we have a cluster of machines?
- Hadoop makes it easy to distribute the calculation
- Purpose-built for parallel calculation
- Just a slight mindset adjustment
Back to Hadoop

- Didn’t we have a cluster of machines?
- Hadoop makes it easy to distribute the calculation
- Purpose-built for parallel calculation
- Just a slight mindset adjustment
- **But a fun one!**
Analysis at Scale

- Now we’re rolling
- Count all tweets: 20+ billion, 5 minutes
- Parallel network calls to FlockDB to compute interest graph aggregates
- Run PageRank across users and interest graph
But...

- Analysis typically in Java
- Single-input, two-stage data flow is rigid
- Projections, filters: custom code
- Joins lengthy, error-prone
- n-stage jobs hard to manage
- Data exploration requires compilation
Three Challenges

- Collecting Data
- Large-Scale Storage and Analysis
- Rapid Learning over Big Data
Pig

- High level language
- Transformations on sets of records
- Process data one step at a time
- Easier than SQL?
Why Pig?

Because I bet you can read the following script

Change this to your big-idea call-outs...
A Real Pig Script

Just for fun... the same calculation in Java next
No, Seriously.
Pig Makes it Easy

- 5% of the code
Pig Makes it Easy

- 5% of the code
- 5% of the dev time
Pig Makes it Easy

- 5% of the code
- 5% of the dev time
- Within 20% of the running time
Pig Makes it Easy

- 5% of the code
- 5% of the dev time
- Within 20% of the running time
- Readable, reusable
Pig Makes it Easy

- 5% of the code
- 5% of the dev time
- Within 20% of the running time
- Readable, reusable
- As Pig improves, your calculations run faster
One Thing I’ve Learned

- It’s easy to answer questions
- It’s hard to ask the right questions
One Thing I’ve Learned

- It’s easy to answer questions
- **It’s hard to ask the right questions**
- Value the system that promotes innovation and iteration
One Thing I’ve Learned

- It’s easy to answer questions
- **It’s hard to ask the right questions**
- Value the system that promotes innovation and iteration
- More minds contributing = more value from your data
Counting Big Data

- How many requests per day?
Counting Big Data

- How many requests per day?
- Average latency? 95% latency?
Counting Big Data

- How many requests per day?
- Average latency? 95% latency?
- Response code distribution per hour?
Counting Big Data

- How many requests per day?
- Average latency? 95% latency?
- Response code distribution per hour?
- Twitter searches per day?
Counting Big Data

- How many requests per day?
- Average latency? 95% latency?
- Response code distribution per hour?
- Twitter searches per day?
- Unique users searching, unique queries?
Counting Big Data

- How many requests per day?
- Average latency? 95% latency?
- Response code distribution per hour?
- Twitter searches per day?
- Unique users searching, unique queries?
- Links tweeted per day? By domain?
Counting Big Data

- How many requests per day?
- Average latency? 95% latency?
- Response code distribution per hour?
- Twitter searches per day?
- Unique users searching, unique queries?
- Links tweeted per day? By domain?
- Geographic distribution of all of the above
Correlating Big Data

- Usage difference for mobile users?
Correlating Big Data

- Usage difference for mobile users?
- ... for users on desktop clients?
Correlating Big Data

› Usage difference for mobile users?
› ... for users on desktop clients?
› ... for users of #newtwitter?
Correlating Big Data

- Usage difference for mobile users?
- ... for users on desktop clients?
- ... for users of newtwitter?
- Cohort analyses
Correlating Big Data

- Usage difference for mobile users?
- ... for users on desktop clients?
- ... for users of #newtwitter?
- Cohort analyses
- What features get users hooked?
Correlating Big Data

- Usage difference for mobile users?
- ... for users on desktop clients?
- ... for users of #newtwitter?
- Cohort analyses
- What features get users hooked?
- What features power Twitter users use often?
Research on Big Data

› What can we tell from a user’s tweets?
Research on Big Data

› What can we tell from a user’s tweets?
› ... from the tweets of their followers?
Research on Big Data

› What can we tell from a user’s tweets?
› ... from the tweets of their followers?
› ... from the tweets of those they follow?
Research on Big Data

› What can we tell from a user’s tweets?
› ... from the tweets of their followers?
› ... from the tweets of those they follow?
› What influences retweets? Depth of the retweet tree?
Research on Big Data

› What can we tell from a user’s tweets?
› ... from the tweets of their followers?
› ... from the tweets of those they follow?
› What influences retweets? Depth of the retweet tree?
› Duplicate detection (spam)
Research on Big Data

› What can we tell from a user’s tweets?
› ... from the tweets of their followers?
› ... from the tweets of those they follow?
› What influences retweets? Depth of the retweet tree?
› Duplicate detection (spam)
› Language detection (search)
Research on Big Data

› What can we tell from a user’s tweets?
› ... from the tweets of their followers?
› ... from the tweets of those they follow?
› What influences retweets? Depth of the retweet tree?
› Duplicate detection (spam)
› Language detection (search)
› Machine learning
Research on Big Data

- What can we tell from a user’s tweets?
- ... from the tweets of their followers?
- ... from the tweets of those they follow?
- What influences retweets? Depth of the retweet tree?
- Duplicate detection (spam)
- Language detection (search)
- Machine learning
- Natural language processing
Diving Deeper

- HBase and building products from Hadoop
- LZO Compression
- Protocol Buffers and Hadoop
- Our analytics-related open source: hadoop-lzo, elephant-bird
- Moving analytics to realtime

http://github.com/kevinweil/hadoop-lzo
http://github.com/kevinweil/elephant-bird
Questions?

Follow me at twitter.com/kevinweil

Change this to your big-idea call-outs...