ECE 574 – Cluster Computing
Lecture 19

Vince Weaver

http://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

13 April 2017

Announcements

- HW#8 was posted
- Don't forget project topics

Google TPU

- Tensor Processing Unit
- ISCA paper – In Datacenter Performance Analysis of a Tensor Processing Unit

Databases

- Machines that store large amounts of data, often optimized for fast retrieval
- Databases
- Relational databases: instead of storing items for each entry together, with every possible field (possibly empty) allocated, instead store each item separately indexed by unique key
- SQL (structured query language)

```
SELECT *
FROM  Book
WHERE price > 100.00
ORDER BY title;
```

- Consistency?
- NoSQL?
- Can have parallel and distributed databases too. It's more difficult with SQL

  - Replication – task runs, making sure all the various copies are kept in sync
  - Duplication – there is a master, and all the others are copies of the master. Users may only change master

- Main memory database – machines with 100TB of RAM?

Big Data

- A buzzword?
- How big is big?
- Terabytes?
- Too big for one machine?
- In general if fits in RAM (¡8GB) or fits on disk (¡10TB) you are better off just using a database or similar

- Once it won't fit on one machine, and you want to use a cluster, things get complicated.

Big Data Tools

- There are various
- Hadoop is one of the more popular

Hadoop

- A distributed filesystem (HDFS)
- A way to run map-reduce jobs

Hadoop

- Apache
- Distributed Processing and Distributed Storage on commodity clusters
- Java based
- Data spread throughout nodes
  Large data sets split up and spread throughout the cluster

- Unlike traditional HPC clusters, code sent *to the nodes* that have data of interest, rather than taking data over network to running code.

- HADOOP common – libraries

- HADOOP YARN – thread scheduling

- Hadoop Distributed File System – HDFS

- Hadoop MapReduce – processing algorithm

- Originally developed at Yahoo by Cutting and Cafarella.

Named after toy elephant.

- Many users. As of 2012 Facebook had 100PB of data, said it grew at 0.5PB/day

Hadoop Distributed Filesystem

`https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html`

- Keeps working in face of hardware failures
- Streaming data access – optimize for bandwidth, not latency
  Relaxes some POSIX assumptions
- Large data sizes – optimized for files of gigabytes to terabytes

- Write-once-read-many – assumption is the data isn't being actively written.

- "Moving computation easier than moving data"

- blocksize and replication factor per-file

- Rack-aware filesystem

- "location awareness" Tries to spread code out multiple copies distributed physically

- Data spread throughout nodes. Default replication value of 3, duplicated three times, twice on same rack and once on different

- Namenode plus cluster of datanodes
- Namenode tracks filenames and locations, keeps entire map in memory
- Datanode stores data. Uses local computer's underlying filesystem. Just blocks of data, makes directories as appropriate but doesn't necessarily have any relationship to the files as seen from within HDFS.

- Communication is over TCP

HDFS Fault Handling

- Datanodes send heartbeats to namenode. When datanodes go missing, marked as dead, no new I/O sent to them. If any files fall below replication level they can be replicated on remaining nodes
- Rebalancing – if disk availability changes files might be moved around
- Integrity – checksums on files to detect corruption
- Namenode is a single point of failure. Keeps the edit log

and fsimage, only syncs at startup

Data Organization

- Data broken up into chunks, default 64MB
- Creating a file does not necessarily allocate a chunk; it is cached locally and only sent out once enough data has accumulated to fill a block
- Replication pipeline: once file created starts being sent in smaller chunks (4kb) and it gets forwarded 1 to 2 to 3 in a pipeline until file in all places.
- Deleting a file does not delete right away, moved to

/trash After configurable time gets deleted from trash and the blocks are marked as free. It can take a while for this to all happen, deletes do not free up space immediately.

- Not a full POSIX filesystem. Writes are slow, and you can't write to an existing file.

Map Reduce

- Originally popularized by Google, but not really used by them anymore (after 2014)
  Jeffrey Dean, Sanjay Ghemawat (2004) MapReduce: Simplified Data Processing on Large Clusters, Google.

- For processing large data sets in parallel on a cluster

- Similar to MPI reduce and scatter operations

- Map() – filters and sorts data into key/value pairs

Stateless, can run in parallel
can contain Combiner() – combines duplicates?

- Reduce() – the various worker nodes process each group in parallel.
  Shuffle() – redistribute data so all common data on same node

- Can do with single processor systems, but not any faster typically. Shines on parallel systems

Map Reduce Example

```
The quick brown fox jumped over the lazy dog.
```

MAP split by key (in this case, number of letters)

```
3: [the, fox, the, dog]
4: [over, lazy]
5: [quick, brown]
6: [jumped]
```

REDUCE each thread/node gets one of these. Reduce might simply count.

```
3: 4
4: 2
5: 2
6: 1
```

Map Reduce Hello World

This is the example they like to use.

Map: key is the word

```
To be or not to be, that is the question.
```

```
to: [1, 1]
be: [1, 1]
or: [1]
not: [1]
that: [1]
is: [1]
```

```
the: [1]
question: [1]
```

Reduce:

```
to: 2
be: 2
or: 1
not: 1
that: 1
is: 1
the: 1
question: 1
```

Real world friends example

- http://stevekrenzel.com/finding-friends-with-ma
- https://www.tutorialspoint.com/hadoop/hadoop_mapreduce.htm

Submitting a Job

- Job:
  Specify input and output on filesystem
  The jar file (java class) of the map and reduce functions
  Job configuration

- Hadoop client sends this to the scheduler

Scheduling

- Each location of system known. Try to run code on same system as data for locality, If not possible, run on one nearby.
- Small cluster has single master node, and multiple worker nodes.
- Hardware does not have to be fault tolerant; if a map/reduce fails it is simply retried again (on another machine)

- You can add/remove hardware at any time

Hadoop Update

Can set up Hadoop on single machine, even the name and data servers. Just download big chunk of Java, have Java and ssh installed.