Distributed Systems

2016 Exam 3 Review

Paul Krzyzanowski

Rutgers University

Fall 2016

Question 1

Unlike the design of Chord, Amazon Dynamo stores the entire list of nodes on each system. Explain the advantage of doing this instead of using a finger table.

Each node knows exactly which node contains the desired key and can forward the query there directly. There is no need for multiple hops.

A finger table may require hops: O(log N) vs. O(1)

Bad answers:

- *Better fault tolerance*
- *Easier to add new nodes*
  (if every node stores the entire list, you still need to update all nodes)

Question 2

You have access to a file of class enrollment lists. Each line contains *{course_number, student_id}*.

Explain how you would use MapReduce to get information on how many classes students take.

For instance, you may discover that 1,495 students are enrolled in 6 courses; 13,077 students are enrolled in 5 courses; 14,946 students are enrolled in 4 courses; and 4,484 students are enrolled in 3 courses.

Explain each map and reduce operation. You may use pseudocode and assume that functions such as sum and count exist. Be sure to state the inputs & outputs of each step.
Hint: you may need more than one iteration

Question 2 (cont.)

You have access to a file of class enrollment lists. Each line contains *{course_number, student_id}*.

First MapReduce: find # of courses each student takes

Map_1:

  input: { *course_number, student_id* }

  output: { key=*student_id*, *1* }

1 for each course per student

Reduce_1:

  input: { student_id, courses[] }

  output: { student_id, sum(courses) } We can also output { 1, sum(courses) }

Second MapReduce: find # of students that take each course count

Map_2:

  input: { student_id, course_count }

  output: { course_count, 1 }

1 for each student with course_count courses

Reduce_2:

  intput: { course_count, students[] }

  output: { course_count, count(students[]) }

Question 2 (cont.)

Input

213	130972375
416	162692062
416	534744968
416	021693896
417	162692062
417	130972375
519	130972375
...	

Output from map

```
key=130972375,  1
key=162692062   1
key=534744968   1
key=021693896   1
key=162692062   1
key=130972375   1
key=130972375   1
…
```

Student 130972375 takes 3 classes

Input to reduce

130972375	1, 1, 1
162692062	1, 1
534744968	1, 1, 1, 1
021693896	1, 1, 1, 1
…	

Output from reduce

130972375	3
162692062	2
534744968	4
021693896	4
…	

Question 2 (cont.)

Input to map = output from reduce

```
130972375   3
162692062   2
534744968   4
021693896   4
…
```

Output from map

```
key=3, 1
key=2, 1
key=4, 1
key=4, 1
…
```

“one student taking 2 courses”

“one student taking 4 courses”

“one student taking 4 courses”

Input to reduce

```
2, {1, 1, 1, 1, ... }
3, {1, 1, 1, 1, 1, … }
4, {1, 1, 1, 1, 1, 1, … }
…
```

$\Rightarrow$

Output from reduce

```
2, 1622
3, 4484
4, 14946
…
```

Question 3

How does Spanner provide consistent lock-free reads of lots of data even if other transactions are modifying some of that data during the read?

Spanner stores multiple timestamped versions in each field.

Snapshot reads allow reading of data whose
    version ≤ transaction start timestamp

Question 4

Explain why each of the following statements is incorrect.

In Pregel, each compute process represents an edge of a graph along with the data that is sent on that edge.

No. Each process represents a vertex

Question 5

Explain why each of the following statements is incorrect.

A shared nothing architecture does not allow two systems to access the same NFS server.

A shared nothing architecture does not allow two systems to share the same block-level storage (e.g., cluster file system).

No credit for simply negating the statement!

Question 6

Explain why each of the following statements is incorrect.

Akamai uses dynamic DNS to look up a URL and return the address of the closest caching server that has that URL in its cache.

1. Akamai uses dynamic DNS to look up domain names, not URLs!
   DNS does not do URL lookups

2. Akamai does not necessarily return the closest server.
   It will return one that is available, not loaded, likely to contain content, …
   and is closest (lowest latency)

Question 7

Explain why each of the following statements is incorrect.

A digital certificate contains a hash that is encrypted with the certificate owner's private key.

Encrypted hash = signature.

A digital certificate is singed by the certification authority (CA), not the owner:

> The hash is encrypted with the CA's private key

Not: encrypted with the certificate owner's public key

Question 7

Explain why each of the following statements is incorrect.

A digital certificate contains a hash that is encrypted with the certificate owner's private key.

Encrypted hash = signature.

A digital certificate is singed by the certification authority (CA), not the owner:

The hash is encrypted with the CA's private key

Not: encrypted with the certificate owner's public key

Question 8

What is an advantage of Amazon Dynamo's *virtual nodes* over simply having each physical machine be a node?

a) Performance is improved because you can create a massive number of virtual nodes.
b) It's cheaper because the node runs on a virtual machine that can do other things as well.
c) A virtual node can be easily migrated to another system.
d) You can shed load from multiple systems when you add a new physical machine..

Adding a new machine adds more virtual nodes that are scattered throughout the ring.

→ Each virtual node takes on some {key, value} sets from another node
On average, these other nodes are spread across multiple physical systems

Question 9

Amazon Dynamo's *optimistic replication*:

a) Uses Paxos to ensure that all replicas contain identical copies.
b) Grabs locks on all replicas of objects while they are being updated.
c) Forces each transaction to check at commit time whether the data was modified by another transaction.
d) Allows some replication to fail, causing some of the replicas to have older versions.

Optimistic replication = it's ok to fail on replication

Dynamo is designed to be always available and eventually consistent.

Question 10

MapReduce can process a lot of data relatively quickly because:

a) Input data is broken up among many map workers.
b) Map workers run in parallel with reduce workers.
c) Reduce workers shrink the amount of data that needs to be processed.
d) All of the above.

MapReduce is a great example of divide-and-conquer.

(b) No. Reduce workers run after every single map worker is done

(c) Only sometimes with multiple iterations of MapReduce. In general, map workers get rid of unnecessary data and reduce workers compute results

Question 11

The *partitioning* function in MapReduce determines:

a) How the original input data is divided into multiple shards.
b) Which reduce worker will work on a specific key.
c) How the map worker parses input data to generate a key.
d) The maximum number of values that will be assigned to each key.

The partitioning function is applied to the {key, value} results from map workers.

Question 12

A column family in Bigtable is:

a) A group of columns within a single table.
b) A group of related columns among multiple tables.
c) A set of columns that are shared across multiple tables.
d) A preconfigured set of columns to make it easier to set up a Bigtable instance.

A column family is a storage unit for multiple columns in a table – usually related in function.

From a performance point, data in a row is retrieved by column families

Question 13

Rows in Bigtable:

a) Are always stored in a sorted order by the row key.
b) Are ordered chronologically; the last row always gets added to the end of the table.
c) Are not sorted but a row can be located by looking up its key in a hash table stored at the master.
d) Are not sorted or searchable but an application can get a list of all rows that contain data for a specific column.

Question 14

Spanner transactions are kept isolated by:

a) Using the TrueTime API to ensure no two transactions start at the same time.
b) Using strict two-phase locking on resources.
c) Using a two-phase commit protocol.
d) Waiting until all previous transactions have committed.

(a) Transactions may start at the same time.

(c) This deals with commits, not isolation during the transaction.

(d) No – but this would also deal with commits rather than isolation

Question 15

To provide external consistency in Spanner, commits are delayed:

a) Until all earlier transactions have released all their locks.
b) All the locks used by the transaction are released.
c) Up to the estimated clock skew between clocks in different locations.
d) Until all earlier transactions have committed.

External consistency = order of transactions reflects their true time order

Commit wait: before a transaction commits, it acquires a commit timestamp:

t = TT.now().latest

t = the latest possible value of the true time across all servers in the system.

Do not release any locks until that time is definitely in the past.

This means waiting until the earliest possible current time on any system is greater than the transaction timestamp:

TT.now().earliest > t

Question 16

In the Pregel framework, the entire job is complete when:

a) Every compute function vote to halt.
b) Every compute function entered an inactive state.
c) There are no incoming messages for any compute function.
d) Every compute function entered an inactive state and has no incoming messages.

Voting to halt == entering an inactive state

BUT:
- A compute function can send a message before voting to halt
- If a compute function receives a message, it is reawakened – placed back into an active state

Question 17

A *combiner* in Pregel:

a) Processes several messages to create a single input to a compute function.
b) Allows the graph to keep global state.
c) Combines multiple edges into a single edge.
d) Combines multiple vertices into a single vertex.

Combiner:

User-defined function applied to a set of messages targeted to the same vertex – creates a single message to that vertex

Aggregator: global variable available to all vertices

Question 18

A Spark resilient distributed dataset (RDD) is fault tolerant because:

a) It is stored as replicas on multiple servers.
b) It is partitioned across many servers in the cluster.
c) It can be regenerated using the transformation that created it.
d) It is cached in memory with a backup on the disk of the servers that created it.

(a) No. An RDD is sharded but generally not replicated.

(b) Yes, it could be partitioned but that does not make it fault tolerant.

(d) Also possible but disk is generally used as a RAM overflow.

RDDs are created by transformations and can be re-created by those same transformations.

Question 19

Which statement best describes Spark's processing?

a) A sequence of transformations followed by an action.
b) A sequence of actions followed by a transformation.
c) An alternating sequence of transformations and actions.
d) An arbitrary mix of transformations or actions.

Transformations read an RDD and create a new RDD.
Example transformations are *map, filter, groupByKey,* and *reduceByKey*.

Actions are operations that evaluate and return a value to the driver program.
Example actions are *reduce, grab samples*, and *write to file*.

Question 20

A System Area Network (SAN) is typically:

a) An overlay network that identifies the nodes that make up the cluster.
b) A dedicated network that is used only for cluster management messages, such as heartbeats.
c) A high bandwidth, high reliability, low latency network used to connect nodes of a cluster.
d) A high-speed network used to connect processors within a multiprocessor system.

High performance, low-latency, high-reliability LAN interconnect for cluster members.

Examples: Infiniband, Myrinet, ethernet with Data Center Bridging

Question 21

A Storage Area Network (SAN) is

a) A network that provides block level access to storage devices via a network.
b) A network dedicated to distributed file system servers, such as NFS servers.
c) A set of connected servers that provide file system services.
d) An overlay network to allow systems in a cluster to identify storage servers.

SAN: network for remote disk access

Question 22

The difference between a clustered file system and a distributed file system is

a) A clustered file system is only made available to a limited set of systems.
b) Distributed file systems may span multiple servers.
c) Clustered file systems provide replication for fault tolerance.
d) All nodes access the clustered file system at the block level.

Each system maintains its own file system and uses a shared block device for storage.

Distributed file system: server(s) maintain the file system and provide file system APIs for remote access (e.g., *read bytes from a file, write bytes to a file, create a file, delete a file*)

Question 23

Checkpointing enables

a) Cold failover.
b) Warm failover.
c) Hot failover.
d) Cascading failover.

Cold failover = application restart

Warm failover = restart but restore state from last checkpoint

Hot failover = backup node is always synchronized and ready to take over

Cascading failover = failover node failing over

Question 24

Akamai's overlay network is primarily designed to:

a) Enable load balancing.
b) Enable load shedding.
c) Provide secure transport of requests within Akamai's systems.
d) Find faster routes than might be available via Internet routing.

Akamai collects statistics about its servers and their availabiliy & connectivity

Core function of Akamai: content caching

Akamai's dynamic DNS enables load balancing & load shedding

Overlay network enables Akamai to find faster routes to origin servers

Question 25

The Diffie-Hellman algorithm was designed to:

a) Allow two parties to come up with a shared secret key while using a public network.
b) Allow one party to use public key cryptography to send a secret key to the other party.
c) Ensure that messages are signed as well as encrypted.
d) Enable an arbitrary number of parties to agree on a common secret key so they all can communicate.

The Diffie-Hellman algorithm allows two parties to create a common key while conversing over an insecure network.

(b) Does not use public key cryptography
(c) Does not use digital signatures
(d) No. Only two parties can come up with a common key.

Question 26

A session key is:

a) A key derived from one user's private key and another user's public key.
b) A randomly generated key designed to be used for one communication session.
c) A key that is used only for signing messages.
d) A key that automatically expires after a certain number of messages have been sent.

Session key = random, throw-away key used for one session.

Question 27

Salt in a password is:

a) A permutation of a password to ensure that it does not match any dictionary words or well-known passwords.
b) A secret transformation of the password performed prior to storing it.
c) Non-secret data added to a password prior to hashing it.
d) An encryption of the password that will require an administrator's secret key to decrypt it.

Salt is data added to a password to thwart attacks based on precomputed hashes

(a) Salt does not permute a password; it adds to it.
(b) It is not secret.
(d) It does not encrypt the password.

Question 28

For Alice to authenticate Bob, she sends him a bunch of bytes that:

a) Bob encrypts with his private key.
b) Bob encrypts with his public key.
c) Bob encrypts with Alice's private key.
d) Bob encrypts with Alice's public key.

(a) Alice can decrypt the result using Bob’s public key to validate that only Bob could have performed the encryption

(b) Anyone can encrypt the data with Bob’s public key.

(c) Bob does not have access to Alice’s private key.

(d) Anyone can encrypt the data with Alice’s public key.

Question 29

A hybrid cryptosystem:

a) Uses two layers of encryption for extra security.
b) Combines the convenience of public key cryptography with the speed of symmetric cryptography.
c) Both encrypts and signs each message to prevent tampering.
d) Alternates among multiple encryption algorithms throughout communication.

Question 30

Tunneling is the technique of

a) Encrypting the data part of a packet before transmitting it.
b) Encrypting the entire packet before transmitting it.
c) Using the data part of a packet to transport another packet.
d) Signing a packet to prevent unauthorized modifications to it.

Tunneling is simply the encapsulation of one packet within the payload of another.

VPNs use tunneling and add encryption and/or signatures.

The End