24.2 Consistent Hashing - What is the benefit of hashing each node (computer) multiple times (instead of just once)?
Answer
Store more data.
Reduces the probability that the storage deviates from the expected value.
Improves the performance when inserting new data into the consistent hash, as the hash values are closer together.
It reduces the amount of data to be moved around (after a node crashes).

not 24.2 Hypercubic Networks (so I should ask this before explaining what they are?) - Which of the following topologies are homogenous?
Answer
Mesh
Fat Tree
Skip list
Torus

24.19 DHT - Which statements on hypercube-based DHTs are true?
Answer
The only possible way for a node to switch to a different hypernode is a dimension change.
Periphery nodes should not know periphery nodes of a neighboring hypernode.
Each node inside a hypernode has a connection to all other hypernodes.
Data is stored only on the periphery nodes.

24.6 Routing a hypercube - Consider routing in a d-dimensional hypercube. How many shortest routes exist when source and target differ by k bits?
Answer
$polynomial(k)$
2^k
$k!$
This is not just a function of k, but also d.

24.1 Advantages of distributed storage - What are advantages of distributed storage over just having one mighty server?
Answer
Latency: Downloads start faster.
Fault-tolerance: No single point of failure.
Search: It's easier to find a node containing the file.
Scalability: Cheaper to add more files.

22.12b What would you check as node? - As a Bitcoin network node
Answer
Inputs are still in UTXO.
Inputs have correct cryptographic signatures.
Input money $=$ output money.
Input was not output of another transaction which was marked as fraud.

22.6 Statements about Algorithm 22.6 - Which statements about algorithm 22.6 are correct?
Answer
When the bank is not reachable, customers can get as much money as they want.
Since it is eventually consistent, a customer cannot withdraw more money than there is on the account.
Only when the bank is reachable, a transaction can be aborted.
During a period when the bank is not reachable, a customer could withdraw from two different ATMs and eventually only one withdrawal would be logged.

22.22 Attacks on Bitcoin - Which of the following attacks on Bitcoin work?
Answer
51% attack: If someone possesses 51% of the mining power on the network, they can mine a majority of the blocks and can therefore control the blockchain.
Denial of Service: An attacker can send a lot of packets to other nodes, making them so busy that they cannot process normal bitcoin transactions. If an attacker manages to hinder a large part of the network from mining, the attacker can execute the above attack even when possessing less than 51% of the computing power.
Spam: An attacker can repeatedly broadcast transactions to itself (transaction from the attacker to the attacker). Like this, blocks will contain a lot of these spam transactions and there's little space left for sensible transactions. The attacker can do this continuously, causing huge wait times for transactions.
Doublespend attack: An attacker owns addresses A and B and mines a block containing a transaction that transfers all money from A to B. The attacker doesn't broadcast the block but runs down to a store and buys a good with money in A: After the attacker broadcasts the transaction from A to the vendor, the vendor hands over the bought product. Only now the attacker broadcasts its mined block. The vendor will never receive the Bitcoins as the transaction is a doublespend.

22.12 What happens to double spends? - What happens if an owner of address A issues two transactions that both want to spend the same input A?
Answer
Both transactions will be executed.
The Bitcoin network will punish account A by deleting all money on account A.
The network will ignore both transactions until the owner of account A has sent a clarification transaction which determines which of the two transactions should be valid.
Not much. The network will choose one of the transactions randomly, and ignore the other transaction.

22.31 Causal consistency - Which statements regarding causal consistency are true?
Answer

In Causal consistency, two writes may be seen in a different order by some nodes, whereas in sequential consistency all nodes see writes in the same order.
Causal consistency is linearizable.
Quiescent consistency implies Causal consistency.
Monotonic write consistency guarantees all writes are serializable

22.21 Receiving blocks - Which statements about Algorithm 22.21 are correct?
Answer
Forks happen daily.
If a new block has $h_b < h_{max}$, all its transactions are ignored.
For a new block b, lines 5-8 of the algorithm will be executed for this block b by all nodes or by no node.
The entire UTXO has to be recomputed for all blocks (from genesis to b_{max}) every time a node accepts a new block.

22.22b Improving blockchain performance - Requiring a PoW for each block leads to scalability issues: It takes a long time and a lot of energy for a transaction to be included on the blockchain. Which of the following increase transaction throughput AND preserve security and decentralization properties?
Answer
Bigger blocks: More transactions in each block.
Faster PoW: Lower the difficulty for solving the PoW.
Avoiding forks: An authority decides which block is mined next. This prevents computing resources to be wasted on forks.
Sharding: Split the blockchain into several subchains"."

not 22.27 Micro payment channel - Which statements regarding the simple micro payment channel setup are true?
Answer
The spender is required to set up a singlesig output for the receiver with the maximum funds the spender is willing to spend.
For every exchange of goods throughout the duration of the channel, the sender has to sign the transaction and broadest it to other nodes such that each transaction will eventually be inserted into the blockchain.
At any point throughout the channel duration, the receiver can close the channel and is guaranteed to receive the bitcoins eventually.
The sender may effectivley close the channel by simply not signing the updated transaction.

20.33 GPS in Phone - Most mobile phones...
Answer
predominantly locate themselves with GPS.
ignore the data in the GPS signals, even without being connected to the internet.
also use WLAN data to position themselves.
use a lot of energy if GPS is turned on.

20.3 Clock sources - How can we keep track of time?
Answer
By observing the power line
By counting the oscillations of atoms
By correlating annual solar patterns from light sensors length of day measurements
By using GPS (Global Positioning System)

20.41 Clock Skew - Which of the following statements are correct?
Answer
Average global skew is always smaller than average local skew.
Average global skew is always larger than average local skew.
With the model assumptions, the local clock skew is a constant.
The global clock skew $\Omega(D)$ holds for an arbitrary number of byzantine nodes.

18.22 What if not signed by dealer? - What if shares were not signed by the dealer?
Answer
No problem.
A byzantine node can trick other nodes into computing a wrong secret.
A byzantine node can still compute the secret.
Why should trust a dealer? Sounds terribly non-distributed...

18.4 What is C - What does the $|C|\geq n^2$ in Algorithm 18.4 imply?
Answer
All nodes have written n coins to the blackboard.
All nodes have written some coins to the blackboard.
All the coins on the blackboard might come from the same node.
All nodes are going to see the same sum.

18.12 Algo = Consensus? - Could the algorithm 18.11 be applied to solve consensus?
Answer
Yes, all nodes accept all the same messages.
Yes, but only a single node should start the algorithm.
No because of validity.
No because of termination.

18.4b Number of Coins read? - What is the maximum amount of coinflips a node might read from the blackboard?
Answer
n^2
$n^2 + 1$

$n^2 + n - 1$
$n^2 + n$

18.4c Number of Byz - How many byzantine nodes can Algorithm 18.4 handle?
Answer
$f = 0$
$f < n/10$
$f = n-1$
$f = n$

18.11 Byz Behavior - Assume a correct node u broadcasts own message msg(u). If all byzantine nodes b broadcast echo(b
Answer
The byzantine nodes are not allowed to do that.
The message can possibly be accepted by a single correct node, but not multiple.
No correct node v will broadcast echo(v,msg(u)') and therefore no correct node will accept msg(u)'.
The message echo(v,msg(u)') can be broadcasted by a correct node v, but no correct node will ever accept msg(u)'.

maybe 18.4d Msg Scheduler? - How should a byzantine scheduler schedule the messages (message content is visible for scheduler)?
Answer
There is nothing a byzantine scheduler can do!
Try to schedule such that the sum stays 0.
Keep some 0 and some 1 messages back, then try to such that the sum stays 0.
If some nodes deterministically go for 0 (in the calling randomized algorithm), then try to schedule 1 messages only.

not 18.4e Crash-Resilient Shared Coin with Blackboard 2 - What is the maximum deviation of sum(C) used for the nodes to make a decision?
Answer
[latex1]
[latex1]
[latex1]

not 18.5 Byzantine Nodes - With cryptographic hash functions h
Answer
compute x such that h(x) = z, where z is smallest possible hash.
forge signatures.
send a different message to every node.

16.25 Final Thoughts - We implicitely made a questionable assumption. Which one?
Answer
All this doesn't work beyond 0 and 1.
We assumed that messages are being delivered in random order.
We only studied crash failures, the real world may be worse!
Consensus is not at all important!

16.9 Bivalent leaf - Assume there is an algorithm with an an initial configuration that leads to a configuration tree that has leaf nodes which are bivalent. Does this system solve consensus? If not
Answer
Solves consensus, all conditions are met.
Agreement is violated.
Termination and/or Validity are violated.
Leaf nodes must be univalent, so this question is a bit questionable.

16.1b Paxos vs. Consensus Properties - Which of the Consensus properties are satisfied by the Paxos protocol?
Answer
Agreement: All nodes decide for the same value
Termination: All nodes terminate in finite time.
Validity: The decision value must be the input value of a node
Paxos has a 4th property: fault-tolerance

16.1 Consensus without Validity? - Can you design a consensus protocol for just agreement and termination (without validity)?
Answer
Yes: Paxos!
Yes: decision = 0.
Yes: All nodes exchange their input in one round, and then decide on the majority opinion.
No, that's impossible.

16.20 What about f = n/2? - What about $f = n/2$? Assume that half the nodes have initial value 0
Answer
Some randomized algorithm can also solve that.
Impossible because the initial configuration is bivalent.
Impossible because of slow messages between the two halves.
Impossible because of the validity condition.

16.21 Make Ben-Or Fast?!? - Can we make Algorithm 16.15 fast by replacing Line 19 with $v_i = 0$?
Answer
No, because of validity.
No, because of agreement.

No, because of termination.
Yes!

not 16.14 FLP Summary - Which of the following statements are true.
Answer
Asynchronous consensus is impossible.
Asynchronous consensus is impossible for $f \leq n/2$.
Asynchronous deterministic consensus is impossible.
Asynchronous non-deterministic consensus is impossible.

not (rather 26) 25.24 Backup views - Assume a correct primary is in view v. Which statements are correct?
Answer
All correct backups are also in view v.
The majority of correct backups are also in view v.
Some correct backups can be up to n views ahead (in any view v' > v & v' < v+n).
Some correct backups can be arbitrary behind (in any view v' < v).

25.3 Realistic Gamma? - What would you consider a realistic γ?
Answer
$\gamma = 0$
$\gamma \approx 1/10$
$\gamma = \alpha$
$\gamma \approx 1/2$

25.2 Does Selfish Mining Happen? - Does Selfish Mining Happen in the Real World?
Answer
Yes.
No.
Bitcoin: No. Other cryptocurrencies: Yes.
We cannot know.

25.18 ETH: Why the Nonce? - Why the nonce?
Answer
The nonce improves the cryptography.
Preventing replay attacks.
For statistics.
The nonce has no purpose and will be removed in ETH 2.0.

23.23 Bitcoin Selfishness? - In Bitcoin
Answer
As a Bitcoin user, why would I include a fee in a transaction?
As a Bitcoin network node, why would I forward transactions and blocks to neighbors?
As a large Bitcoin mining pool, why would I always announce a block immediately?
As a Bitcoin hodler" (holder for the rest of us) 0

23.5 Nash Equilibrium - Which of the following statements about the Nash Equilibrium are correct?
Answer
Every game has at least one Nash Equilibrium.
A game can have multiple Nash Equilibria.
In every Nash Equilibrium at least one player achieves the highest possible payoff.
If every player plays a dominant strategy, then this is a Nash Equilibrium.

23.10 Price of Anarchy - Which of the following statements are correct?
Answer
The Optimistic Price of Anarchy of Selfish Caching is always 1.
The Price of Anarchy (PoA) being 1 means that every Nash Equilibrium is a Social Optimum (SO).
The Price of Anarchy can never be bigger than n.
The Optimistic Price of Anarchy can never be bigger than n.

23.5b Iterated Prisoner's Dilemma - Which strategy usually wins in the iterated prisoner's dilemma?
Answer
Always Defect. After all it's the dominant strategy.
Start with cooperate. Then play whatever your partner played in the last round.
Start with cooperate. Then cooperate as long as your partner cooperates. Once your partner defects, always defect.
Previous strategy, but add a bit of randomness. It's not good if you are too predictable.

23.17 Selling the Exam! - How much would you bid for a garanteed 6.0 in the exam?
Answer

not 23.7 Nash equilibrium for selfish caching (not 4 answers... sigh) - Is the algorithm 23.7 truthful?
Answer
No, a node can lie about its demand and reduce the cost for the node
Yes, the node can not gain anything by lying about its demand

not 23.2 prisoners dilemma - Which statements about the prisoner's dilemma are true?
Answer

Choosing a non-dominant strategy never results in a better outcome for the player, regardless of the strategy the other player chooses.
Over multiple iterations of the prisoner's dilemma, always defecting is remains the dominant strategy.
Assume the police have strong evidence that player u has committed a crime such that player v gains no benefit for defecting. Now the Nash equilibrium overlaps with the social optimum.

not 21.20 B-Grid Quorum systems (wrong format) - In wich scenario does a B-Grid Quorum system fail?
Answer
A whole band fails.
In each band a mini-column fails.
In every band one element of a single mini-column fails
In a single band an element in each mini-column fails
A whole column fails

not 21.4 Work and load example (all quorums = 1
Answer
The access strategy inducing the minimal load on the system is choosing a quorum uniformly at random.
All access strategies induce the same amount of work on the system.
The load on the system is \[n/q\], where \[q\] is the amount of quorums. The work on the system is \[n\]. This holds for any uniform system.
The described quorum system does not exist.

21.3 Fault tolerance - Assuming we want a quorum system with uniform work for all quorums. Which of the following scenarios would increase the resilience of a Quorum system?
Answer
Create a system with many smaller quorums.
Create a system with fewer, but larger quorums.
Creating a system designed to maximize the overlap between any two quorums.
Upgrading the system from a singleton quorum system to a majority quorum system.

21.12 Concurrent locking strategy - To avoid deadlocks clients lock the servers sequentially based on an ordering of their identifiers and restart on a failed lock attempt. Does using tickets instead of locks guarantee that at least one node is always making progress?
Answer
Yes, the client has to wait for the majority of the quorum for confirmation.
No, tickets can still cause deadlocks.
Yes, the client has to wait for a response from all nodes in the quorum to make progress
No, tickets solve the issue of deadlocks, but can still lead to starvation.

TODO: all questions today I didn't use them in 2020 because eduapp down. 21.1 Correct Quorum Systems - Which of the following are valid quorum systems?
Answer
A system consisting of all possible node subsets that contain the node with ID 1.
Any minimal system with at least 3 quorums containing at least \[n/2\] nodes.
Any minimal system where each node is part of exactly 2 quorums.
Any minimal system with \[n\] nodes and more than 2 quorums, where one quorum has cardinality \[n-1\]. (Roger disagrees with this answer.)

not 21.4 Byzantine Quorum systems (wrong format) - Assume we have an f-masking Byzantine quorum system. A client has successfully updated the state of the servers in a Quorum Q1. Is Q1 able to propagate the updates to another Quorum system Q2?
Answer
Yes, we can guarantee that the overlap between Q1 and Q2 is at least \[2f + 1\], so the correct nodes outnumber the byzantine nodes in Q2.
No, in the Quorum Q2 the set of out-of-date nodes and byzantine nodes together may outnumber the set of up-to-date correct nodes.

19.25 Lamport clocks are ... - Which statements about Lamport clocks are correct?
Answer
Lamport clocks are easy to implement
Lamport clocks are also strong logical clocks
Lamport clocks are not strong logical clocks
Strong logical clocks are impossible to build

19.9 Sequential consistency vs linearizability - Select the correct implication between sequential consistency and linearizability.
Answer
Sequential consistency $\implies$ linearizability
Linearizability $\implies$ sequential consistency
Sequential consistency $\iff$ linearizability
Sequential consistency and linearizability are independent

19.11 Quiescent consistency vs linearizability - Select the correct implication between quiescent consistency and linearizability.
Answer
Quiescent consistency $\implies$ linearizability
Linearizability $\implies$ quiescent consistency
Quiescent consistency $\iff$ linearizability
Quiescent consistency and linearizability are independent

19.16 Composable Models - Which of the following models are composable?
Answer
Linearizability
Sequential Consistency
Quiescent Consistency

None of the above

19.2 What can o be? - After all these operations
Answer
$o = 16$
$o \in \{15,16\}$
$o \in \{14,15,16\}$
$o \in \{13,15,16,17\}$

19.20 Happened-Before - Happened-before is pretty much the same as ...
Answer
Linearizability
Sequential Consistency
Quiescent Consistency
It's different from these three

19.34 Microservice Advantages - Which of the following statements are advantages of Microservices?
Answer
Easier debugging and optimizing
Better fault tolerance
Less network communication
Better scalability

not19.13 Sequential vs quiescent consistency - Select the correct implication between sequential consistency and quiescent consistency.
Answer
Sequential consistency $\implies$ quiescent consistency
Quiescent consistency $\implies$ sequential consistency
Sequential consistency $\iff$ quiescent consistency
Sequential consistency and quiescent consistency do not imply one another

19.12 Relation of Consistency Models - Which of the given statements are correct?
Answer
Sequential Consistency $\land$ Quiescent Consistency $\implies$ Linearizability
$C_1 \implies C_2$ means that C_1 is cheaper and easier to implement
Sequential Consistency $\implies$ Quiescent Consistency
Quiescent Consistency $\implies$ Sequential Consistency

19.41 Algorithm comparison - Which of the following statements are advantages of the Token-Based Algorithm over the Distributed Algorithm?
Answer
Lower message overhead
Guarantees in-order fairness
More scalable: every node doesn't need to know all nodes participating
No signle point of failure

17.28 Precomputed Common Bitstring? - Does Algorithm 17.28 actually work?
Answer
Yes. This is exactly the same scenario as with the random oracle.
Yes. After all a scheduler is independent of some precomputed random bitstring.
No. This algorithm is now completely deterministic, so it cannot work.
No. This algorithm cannot work because a byzantine scheduler can always schedule the wrong messages. If the next random bit is a 0, the scheduler will make sure that some nodes will deterministically go for a 1.

17.5 All-Same Validity - Which statements are correct for all-same validity?
Answer
If all nodes start with input 1 the decision value has to be 1.
If $f+1$ nodes start with input 1, the decision value has to be 1.
If $n-f-1$ nodes start with input 0, the decision value can be 0 or 1.
If a byzantine node starts with input 1 and all other nodes with input 0, the decision value can be 0 or 1.

17.21 Assessing Ben-Or - How would you assess Algorithm 17.21?
Answer
It is perfect! All our problems are solved.
We now know that there is no need for predefined kings.
While expected runtime is fine, f is unacceptably small.
It is bad. There is no substantial improvement over the King algorithm.

17.11 f=1 with all-same-validity - Can we change Algorithm 17.9 to achieve All-Same-Validity
Answer
No.
Yes: Change Line 8 to choose the value proposed by most nodes (in case of a tie: smallest)
Yes: Change Line 8 to choose value proposed by at least 2 nodes (in case of a tie: smallest)
Yes: But it's more complicated than just changing Line 8.

17.19 Who decided on the kings? - How did we decide on the kings?
Answer
The kings must be known before the algorithm starts.

We simply use the same king all the time.
If the nodes are numbered 1,2,3,..., then we simply use this order as kings.
We can elect the king of phase i+1 at the end of phase i.

17.7 Min. nodes for f=1 (sync) - What is the minimum number of nodes needed to be able to reach byzantine agreement with all-same validity for $f=1$?
Answer
2
3
4
5

15.15b Multiple Commands - Does Paxos support multiple commands (state machine replication)?
Answer
Paxos (Algorithm 15.13) already supports multiple commands. What are you talking about?
Every command gets a number, and we run a Paxos instance for command 1, 2, 3, ...
Same as above, but instead of sequence numbers we refer to a previous command.
Paxos cannot be used for state machine replication.

15.5 Client-Server Algorithm with Acks - Assuming variable message delay
Answer
One client and one server
One client and multiple servers
Multiple clients and one server
Multiple clients and multiple servers

15.15 Paxos Statements - Which of the following statements about the Paxs protocol are correct?
Answer
The Paxos protocol always terminates successfully
At any point a client can decide to abort and get a new ticket
New tickets can only be issued if all previous tickets have been returned
Paxos cannot make progress if half or more of the servers crash

14 Why study CS? - Why did you study Computer Science / Computing?
Answer
I have been a computer nerd all my life, so studying it was pretty obvious.
Computers are everywhere, it's the new math.
There is so much progress in this field, every few years there is something completely new.
I heard that you can earn good money with this degree.

15.10 Two Phase Interference - Multiple clients concurrently try to get all locks. How would you fix the potential deadlock?
Answer
If a client does not get all locks, the client aborts, returns the already acquired locks, and then tries again.
Same as above, but before trying again wait some random time period.
If a client \(u\) asks a server for the lock but the server has given the lock already to another client \(v\), the server will inform \(u\) to talk to \(v\) directly.
Forget about these locks. They are only trouble!

15.13 Paxos Point of No Return - When is the "point of no return" in Paxos? (Point of no return = time when the command cannot be change
Answer
Line 16: At least one server stores the command.
Line 16: All servers store the command.
Line 20: The client sends execute.
None of the above.

not 15.10 Two-Phase Protocol - To achieve progress in the Two-Phase Protocol how many servers need to be responsive?
Answer
only one server
at least a majority of servers
all but one server
all servers

not 15.13 Paxos - Running a single instance of the Paxos algorithm can be used to:
Answer
Execute one command on the majority of servers
Execute multiple commands in the same order on the majority of servers
Execute multiple commands in any order on the majority of servers

not 15.13 Paxos Majority - In Paxos clients are requiring an answer from the majority of the servers. Instead a client can also require a different percentage of servers to answer. For which percentage would Paxos still be correct?
Answer
25.10%
34%
55%
70%