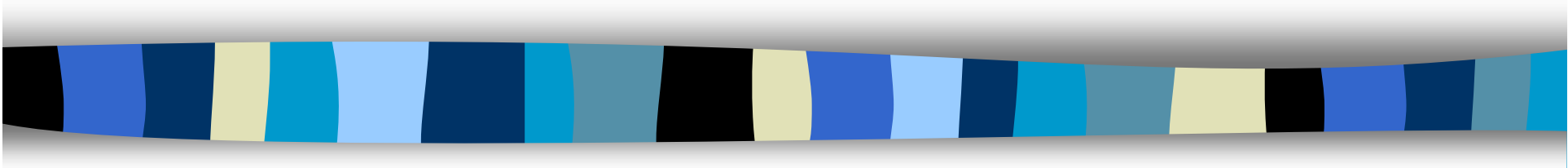


6CCS3AIN AI Reasoning & Decision-Making

Consensus Mechanisms

Week 9 — Part A — Introduction



Peter McBurney
Department of Informatics
King's College London
London

peter.mcburney@kcl.ac.uk

Week 9
30 November 2020

Consensus Mechanisms

- This week, we are going to look at some different mechanisms by which a collection of agents can reach consensus on some question.
- There are three parts to the lecture:
 - Part A is an Introduction given by me
 - Part B is Voter Models (aka Flag Colouring) given by Dr Kohan Marzagão
 - Part C is on Consensus in Blockchain, and is given by me.
- We will touch on some of the models and mathematics involved.



Image: Charley Harper

Intermission

First, I ask you to pause this presentation and look at the following video, about some small robots called Kilobots:

https://www.youtube.com/watch?v=rdM_xJkfn6k

Kilobot robots are 3.3cm tall and were developed from 2010 by Radhika Nagpal & Michael Rubenstein at Harvard University. They were intended to be low-cost robots for the study of swarm robotics. They can be purchased from K-Team in Switzerland.



Image: Charley Harper



Assumptions I

- We have multiple agents or computational entities
- Each agent is programmed with the same program
- Each agent is connected to zero or more others
 - either they are linked together, or they can see each other
- So each agent has a local neighbourhood with a finite set of neighbours
- We can represent this situation as a graph
 - with agents as nodes, and “the connections” between them as edges
 - We use the terms “agent” and “node” interchangeably.



Assumptions 2

- We assume each agent can take a finite number of states (often called “colours”).
 - For example, they can be either Blue or Red
- Interactions proceed in a series of rounds (or editions)
 - Each round, each agent has to decide what state (colour) to be in
 - Agents make their decision according to an algorithm (which could be random)
 - All agents are using the same algorithm
 - The algorithm may first check the colours of the neighbours in the previous round.
- We assume the agents share a common goal to achieve an overall configuration of states, eg:
 - All nodes have the same colour
 - No two adjoining nodes have the same colour
 - The colours alternate (if the nodes are in line or a circle),



Questions a computer scientist would naturally ask

- If each agent can have a finite number of colours, will they ever converge to the desired configuration?
- Is convergence possible from a random start?
- If convergence is possible, what is the probability of convergence?
- If convergence is possible, how fast will the nodes converge?
- Are deadlocks possible?
- Are cycles possible?
- Can convergence be facilitated from particular starting configurations?

A problem domain with similar questions: Rubik's Cube

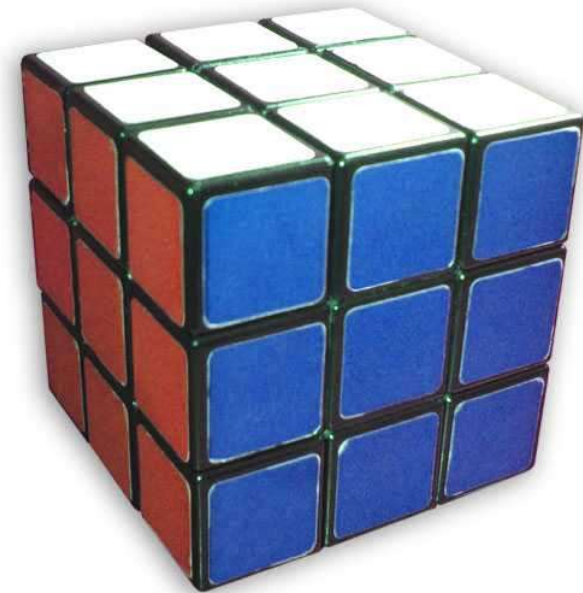
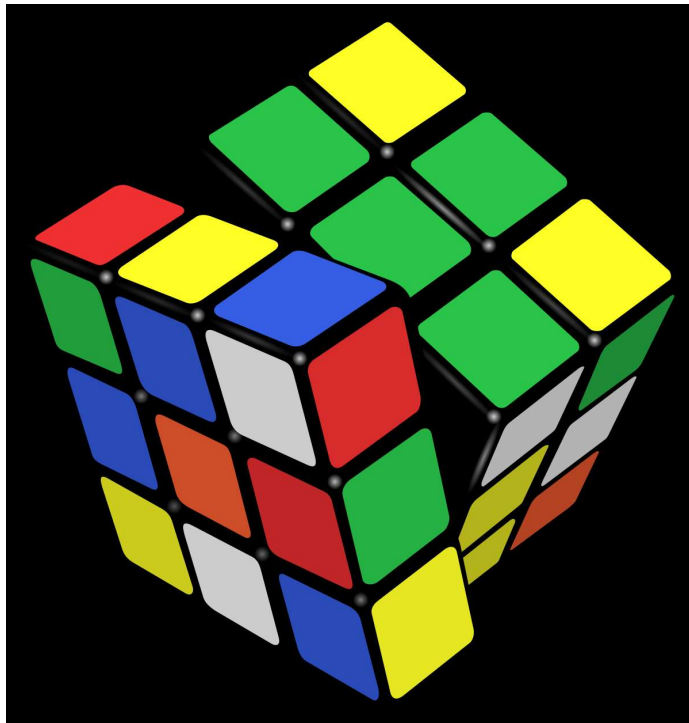


Image source: Wikipedia



Questions an agents person would naturally ask

- What happens if one or more agents are not following the common algorithm?
- For instance, what if one agent is malicious and does not wish to achieve the common goal (of a particular configuration)?
- What if one agent has the same shared goal but has buggy code?
 - It is often difficult to distinguish malice from bugs when we can only observe the agent's external behaviour
- What if one or more agents are temporarily offline?
- Can the agents protect themselves from a malicious or buggy agent? Can the other agents isolate the influence of the dissident agent? Can the other agents even identify dissident behaviour?

Applications I

- Robot bucket brigades
 - Each robot has to be either giving or receiving a bucket at each round
 - So the desired configuration is alternating colours.



Bucket Brigade, Dresden, Germany
After WW II

Source: Wikipedia

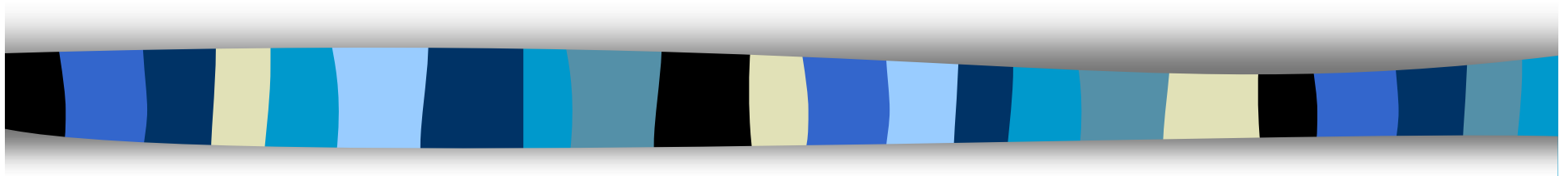
Applications 2

- Consensus in distributed systems (eg, blockchains)
 - We will talk about this in Lecture 9 Part C
- Designing and managing swarms
 - Birds in flocks and bees in swarms seem to continually adjust their speed and trajectory according to the birds they can see around them
 - We can create similar algorithms for swarms of autonomous vehicles, such as flying drones or convoys of ships or convoys of road vehicles.



Image Credit: Rolls Royce Blue Ocean

Thankyou!



6CCS3AIN: Artificial Intelligence Reasoning and Decision Making

Week 9: Consensus Mechanisms

Part B: Voter Model

David Kohan Marzagão

david.kohan@kcl.ac.uk

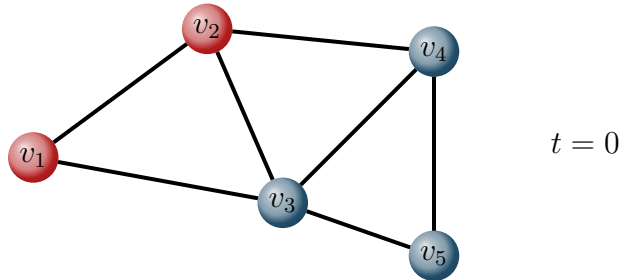
Department of Informatics

King's College London

(version 1.1)

Motivation

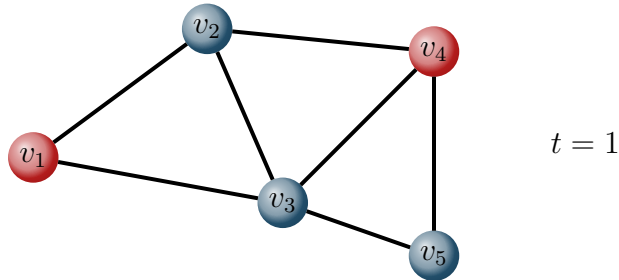
Motivational Example



- Consider a graph such as the one above.
 - Nodes in this graph represent agents.
 - Colour of nodes represent their current opinion. (it may change over time!)
 - Edges represent what other agents a node sees.
- Their goal: to reach consensus. (i.e., all agents with same colour)

Idea They can learn from their

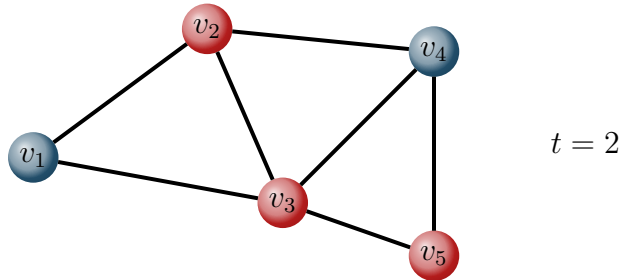
Motivational Example



- Consider a graph such as the one above.
 - Nodes in this graph represent agents.
 - Colour of nodes represent their current opinion. (it may change over time!)
 - Edges represent what other agents a node sees.
- Their goal: to reach consensus. (i.e., all agents with same colour)

Idea They can learn from their neighbours

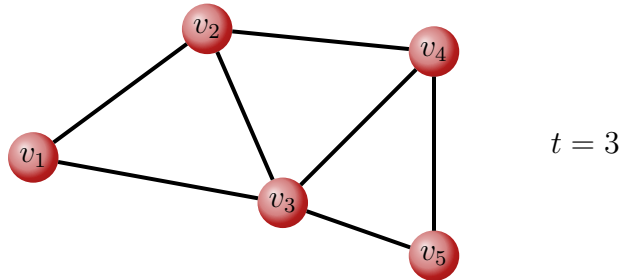
Motivational Example



- Consider a graph such as the one above.
 - Nodes in this graph represent agents.
 - Colour of nodes represent their current opinion. (it may change over time!)
 - Edges represent which other agents a node sees.
- Their goal: to reach consensus. (i.e., all agents with same colour)

Idea They can learn from their neighbours over

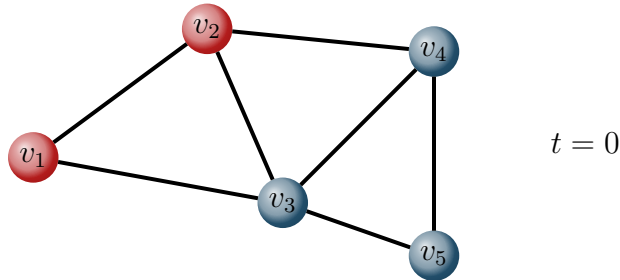
Motivational Example



- Consider a graph such as the one above.
 - Nodes in this graph represent agents.
 - Colour of nodes represent their current opinion. (it may change over time!)
 - Edges represent what other agents a node sees.
- Their goal: to reach consensus. (i.e., all agents with same colour)

Idea They can learn from their neighbours over time.

Motivational Example



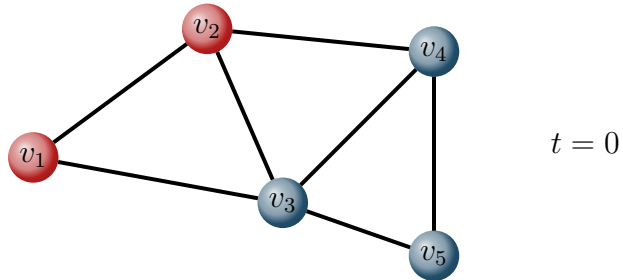
- There are many algorithms that can be considered in this case. Today, we are focusing on the following, usually called **voter model**.
 - The process happens in **rounds**, or time-steps, denoted by t .
 - At each round t , each agent v **selects** one of its neighbours u uniformly at random.
 - Then, v **adopts** u 's colours for next round.

To avoid ambiguity, we consider that they all change at the same time at the end of round t .

Some Notation

- First we assume there is a time variable $t \in T$ that represent rounds. In our case, T is the set of non-negative integers $0, 1, \dots$.
- We have an undirected connected graph $G = (V, E)$.
 - An edge (v, u) indicates that v sees u and u sees v .
 - The number of neighbours of v is denoted by $\deg(v)$.
- A set of colours or opinions X . (in our example, $X = \{b, r\}$)
- For each time $t \in T$, the current state of the process is described by a function $S_t : V \rightarrow X$.
- We are interested in the consensus states. When, for a given t , we have $S_t(v) = b$ for all $v \in V$, we say $S_t = \text{blue}$. Analogously for red, we have $S_t = \text{red}$.
 - In these cases, we say the processes **ends**, and we denote this by saying that $S_{\text{end}} = \text{blue}$, or $S_{\text{end}} = \text{red}$.

Examples of update rules



- 1) What is $Pr(S_1(v_1) = r)$?
- 2) What is $Pr(S_1(v_4) = b)$?
- 3) What is $Pr(S_1(v_5) = r)$?
- 4) What is $Pr(S_1 = \text{blue})$?
- 5) What is $Pr(S_1 = \text{red})$?

Questions

- Is consensus stable? I.e., once consensus is reached, will it be maintained indefinitely?
In other words,

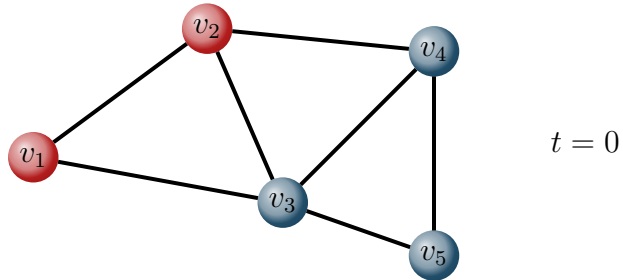
$$S_{t_0} = \text{blue} \quad \Rightarrow \quad S_t = \text{blue, for all } t > t_0?$$

same for red.

- Can the process ever get stuck and never find a consensus state? (exercise!)
- Imagine you are not an agent in this process, but you have the power to flip the colour of an agent before anyone makes a decision in a given round. Which agent is the most influential, i.e., the best choice?
- And our key point:

Question Given an initial configuration S_0 , what is the probability that the game ends in blue consensus (same for red)?

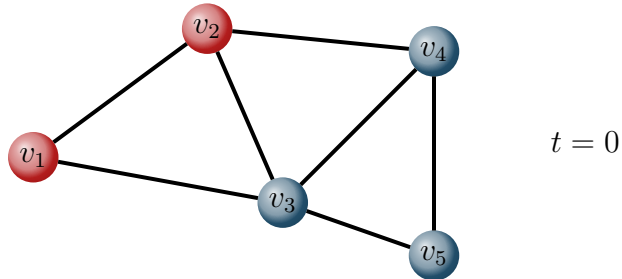
Probabilities of Consensus



- What counts for an opinion's advantage?

Option (1) The number of nodes of a given colour.

Probabilities of Consensus



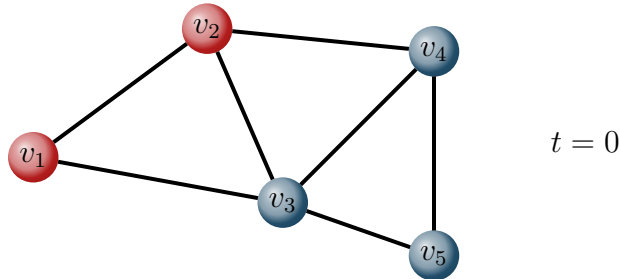
- What counts for an opinion's advantage?

Option (1) The number of nodes of a given colour.

X

Option (2) The sum of degrees of nodes with a given colour.

Probabilities of Consensus



- What counts for an opinion's advantage?

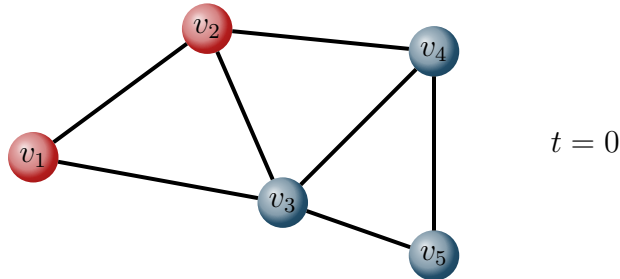
Option (1) The number of nodes of a given colour.

✗

Option (2) The sum of degrees of nodes with a given colour.

✓

Probabilities of Consensus



- What counts for an opinion's advantage?

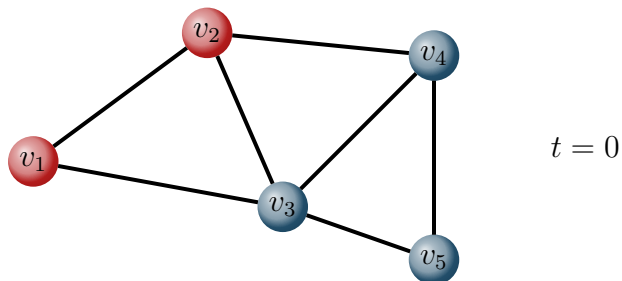
(proof in extra material!)

Theorem Let $G = (V, E)$ be a graph that do not allow deadlocks in a voter model process, i.e., a consensus is reached with probability 1. Given an initial configuration $S_0 = s$, the probability of blue winning is given by

$$Pr(S_{end} = \text{blue} \mid S_0 = s) = \sum_{v \in V, s(v)=b} \frac{\deg(v)}{2|E|}$$

analogously for red.

Probabilities of Consensus - Example



- We apply the theorem to the example above.

$$Pr(S_{end} = \text{blue} \mid S_0 = s) = \frac{4 + 3 + 2}{14} = \frac{9}{14} \approx 64\%$$

$$Pr(S_{end} = \text{red} \mid S_0 = s) = \frac{2 + 3}{14} = \frac{5}{14} \approx 36\%$$

Some easy and not so easy follow up questions

- Can we say something about the expected time this will take to converge to consensus? (see *Hassin and Peleg, 2001* and subsequent work)
- What happens if the graph is not undirected but instead a directed graph with weighted edges? (see *Linear Voting Model Cooper and Rivera, 2016*)
 - We consider that, in this case, the weight of the edge (v, u) represents the probability that v copies colours of u in a given round.
- What is there is some sort of bias toward a colour? For example, if when an agent has a blue and a red neighbour, they may have a higher chance of choosing, say, colour blue. (see *Moran Processes*)

Why not implement this yourself and check if our probabilities check out? Once you are there, make sure to calculate how many rounds it takes on average.

The End

Some references

- [Cooper and Rivera, 2016] Cooper, C. and Rivera, N. (2016). The Linear Voting Model: Consensus and Duality.
- [Hassin and Peleg, 2001] Hassin, Y. and Peleg, D. (2001). Distributed probabilistic polling and applications to proportionate agreement. *Information and Computation*, 171(2):248–268.

6CCS3AIN - Week 9

Part B: Voter Model

Proof of Theorem 1 (version 1.0)

1 Proof of Theorem 1

Recall Theorem 1 from the lecture that we aim to prove:

Theorem 1.1 *Let $G = (V, E)$ be a graph that do not allow deadlocks in a voter model process, i.e., a consensus is reached with probability 1.*

Given an initial configuration $S_0 = s$, the probability of blue winning is given by

$$Pr(S_{end} = blue \mid S_0 = s) = \sum_{v \in V, s(v)=b} \frac{\deg(v)}{2|E|}$$

analogously for red.

Proof. Consider the random variable given by the sum of degrees of nodes of a given colour, say blue, at a given round t . Formally,

$$Y_t = \sum_{v \in V, s(v)=b} \frac{\deg(v)}{2|E|}$$

The proof relies on the fact that this quantity does not change, on average, from one round to the next. In other words, if we calculate all the possibilities of Y_1 given the configuration at round $t = 0$, and weight them by their respective probabilities, their sum would amount to Y_0 . We show that for a given round t ,

$$\mathbb{E}(Y_{t+1} \mid S_t = s) = \mathbb{E} \left(\sum_{v \in V, S_{t+1}(v)=b} \frac{\deg(v)}{2|E|} \mid S_t = s \right)$$

Note that $\mathbb{E}(X \mid Z = z)$ stands for the conditional expectation of X given that we know that $Z = z$. Because the choice of a node is independent of the choice of other (although dependent on their current colours, of course), we can write the expected value above as a sum for each node v .

$$\begin{aligned} \mathbb{E}(Y_{t+1} \mid S_t = s) &= \frac{1}{2|E|} \left(\sum_{v \in V} Pr(S_{t+1}(v) = b \mid S_t = s) \deg v \right) \\ &= \frac{1}{2|E|} \left(\sum_{v \in V} \frac{\text{blue-neighbours}_t(v)}{\deg v} \deg v \right) \end{aligned}$$

where $\text{blue-neighbours}_t(v)$ is number of blue neighbours of v at round t , i.e., the number of elements in the set $\{u \mid u = N(v), S_t(u) = b\}$, where $N(v)$ is the set of neighbours of v .

We can now see that $\deg v$ cancels and we arrive at the sum of blue neighbours for all nodes v . Well, overall that is equivalent to counting each blue node u as many as $\deg u$ times, and thus

$$\mathbb{E}(Y_{t+1} \mid S_t = s) = \sum_{u \in V, S_t(u)=b} \frac{\deg(u)}{2|E|} = Y_t$$

Now that we have shown the variable Y does not change in expectation from one round to the next, we may conjecture that this behaviour carries over for as many rounds as we like. Indeed, Optimal Stopping Theorem guarantees that

$$\mathbb{E}(Y_{end} \mid S_0 = s) = \mathbb{E}(Y_0 \mid S_0 = s) = Y_0 = \sum_{v \in V, s(v)=b} \frac{\deg(v)}{2|E|} \quad (1)$$

On the other hand, Y_{end} can be only two values. Since S_{end} represents the end of the game, and thus a consensus, Y_{end} can either be zero (consensus in red), or 1 (consensus in blue). The expected value of $\mathbb{E}(Y_{end} \mid S_0)$, by definition, is the value the variable can take multiplied by the probability of them happening, thus:

$$\mathbb{E}(Y_{end} \mid S_0 = s) = 0 \times Pr(S_{end} = \text{red} \mid S_0 = s) + 1 \times Pr(S_{end} = \text{blue} \mid S_0 = s) \quad (2)$$

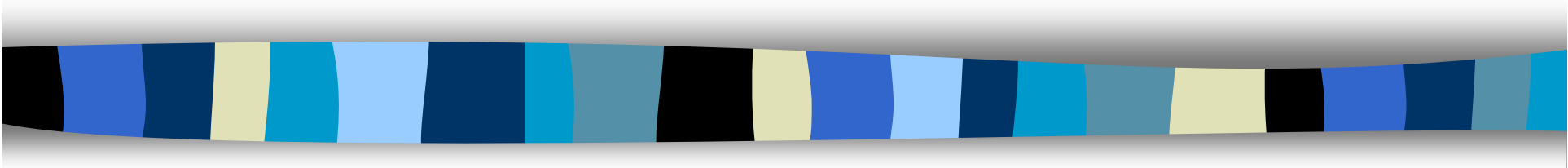
Putting Equations 1 and 2 together

$$Pr(S_{end} = \text{blue} \mid S_0 = s) = \sum_{v \in V, s(v)=b} \frac{\deg(v)}{2|E|}$$

As we wanted. ■

6CCS3AIN AI Reasoning & Decision-Making: Consensus Mechanisms

Week 9 — Part C — Consensus in Blockchain



Peter McBurney
Department of Informatics
King's College London
London

peter.mcburney@kcl.ac.uk

Week 9
30 November 2020



Introduction

- This is Part C of the lecture for Week 9.
- The topic for this week is consensus mechanisms, which are an important part of automated decision-making in decentralized and distributed systems.
- In this lecture, we look at the type of mechanisms used to reach consensus in Blockchain systems, as in the Bitcoin Cryptocurrency.
 - We begin with a few words about Bitcoin and Blockchain.
- The material in this Part C is intended to show you an actual and very significant application of these ideas about automated decision-making in decentralized networks of autonomous agents.
 - The details of the Bitcoin Blockchain are not examinable in this course.

Bitcoin (BTC)

- Bitcoin was conceived in 2008, and was deployed in 2009.
 - Inventor was "Satoshi Nakamoto" (a pseudonym)
 - This is the world's first decentralized electronic currency
- Key problem of an electronic currency:
 - How to ensure there is no double-spending of currency
 - Prior solution: Have a central node in charge of allocation
 - Weakness: Central node may be corrupt or vulnerable to attack
- Bitcoin's solution:
 - Have the allocations of currency witnessed and approved by all nodes in the network.
 - And chaining of data makes it impossible to change past data surreptitiously.





Blockchain

- The underlying technology was called Blockchain
- There was no new invention, just a clever combination of previous CS technologies
 - Peer-to-peer (P2P) network
 - Consensus Protocol for automated collective decision-making
 - Proof-of-Work to enable appropriate economic incentives
 - Cryptography for authentication of participants (PKI – Public Key Infrastructure)
- Basic atoms: Transactions (TXs) shifting allocation of some currency from one node to another node
- Transactions grouped into blocks, then blocks chained together
 - A new block about every 10 minutes
- Now part of Distributed Ledger Technologies (DLT).



Economic incentives

- The designers of the system want to incentivize participation
 - Processing transactions takes some work
- So the system rewards uploading blocks
 - The first node to do so correctly is rewarded with some BTC
- But the network is open and pseudonymous
 - Participation is open and is under a BTC address (no real names or ID info)
 - So a malicious node could join multiple times
 - Possibility of a Sybil attack (multiple-identity attack)
 - Sybil is not to be confused with sibyl (an oracle)
- So the network needs to discourage this
 - To successfully upload a new block, the node has to do some additional processing work
 - This involves solving a math puzzle - “Proof-of-Work” (PoW)
 - The puzzle is hard to solve but easy to check if correct.



Four parts of decentralized consensus

- A Independent verification of each transaction, by every full node
- B Independent aggregation of those transactions into new blocks by mining nodes
together with demonstrated computation through a Proof-of-Work algorithm
- C Independent verification of the new blocks by every node and assembly into a chain
- D Independent selection, by every node, of the chain with the most cumulative computation demonstrated through Proof-of-Work.



The high-level process #1

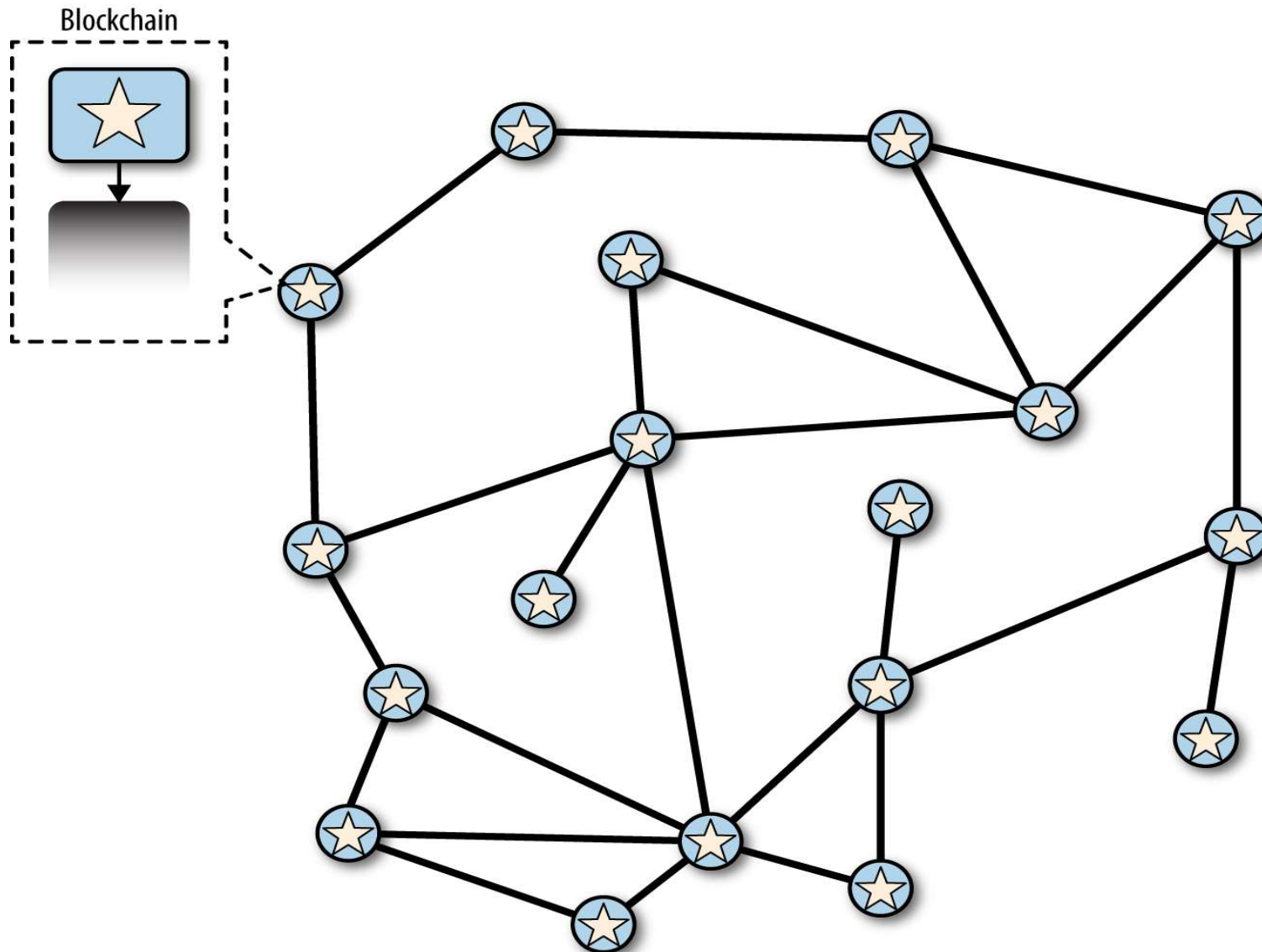
- We have a P2P network
 - Currently about 11,300 nodes
 - No one is in charge
 - All full nodes have the same software
- A node issues a potential transaction (TX)
 - Sent to all nodes in its neighbourhood (those it knows about)
- Each receiving node checks the potential TX for syntax and validity
- If validated, the receiving node sends the TX to all its neighbours, and so on
 - If the same TX is received more than once, subsequent receipts are ignored.



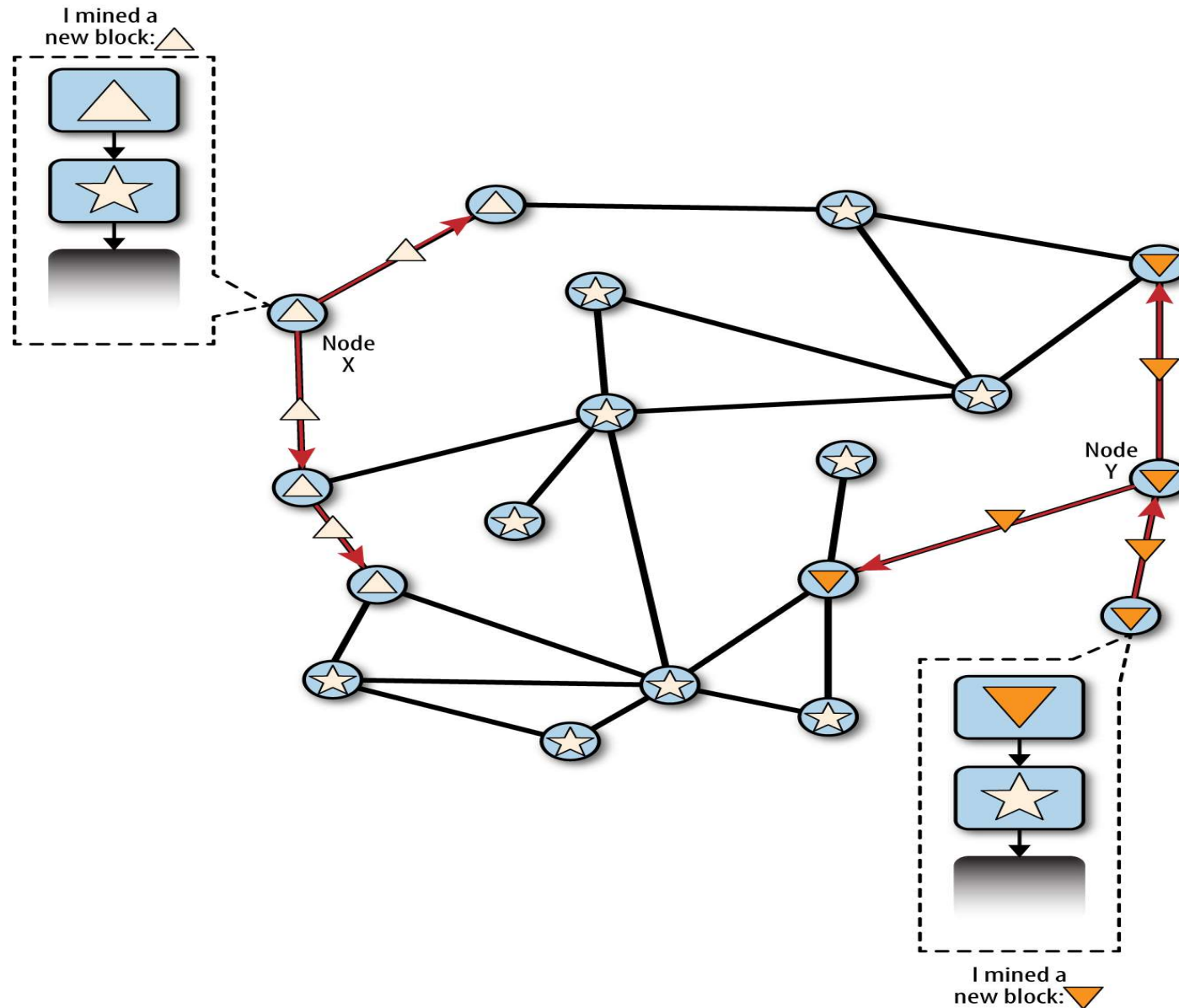
The high-level process #2

- Full nodes try to assemble a new block
 - They concatenate new TXs together
- Full nodes try to solve the math puzzle (by doing work)
 - If they solve it, they add the solution (the Proof-of-Work) to the block header
- They then send the candidate block to their neighbours
- Each receiving node validates the candidate block and adds it to its copy of the existing chain, then sends to its neighbours
- When faced with competing blocks on the chain, decides to accept the chain with the greatest cumulative proof of work.
 - This is the decision to adopt a particular state (or colour)
 - This is called The Longest-Chain-Rule.

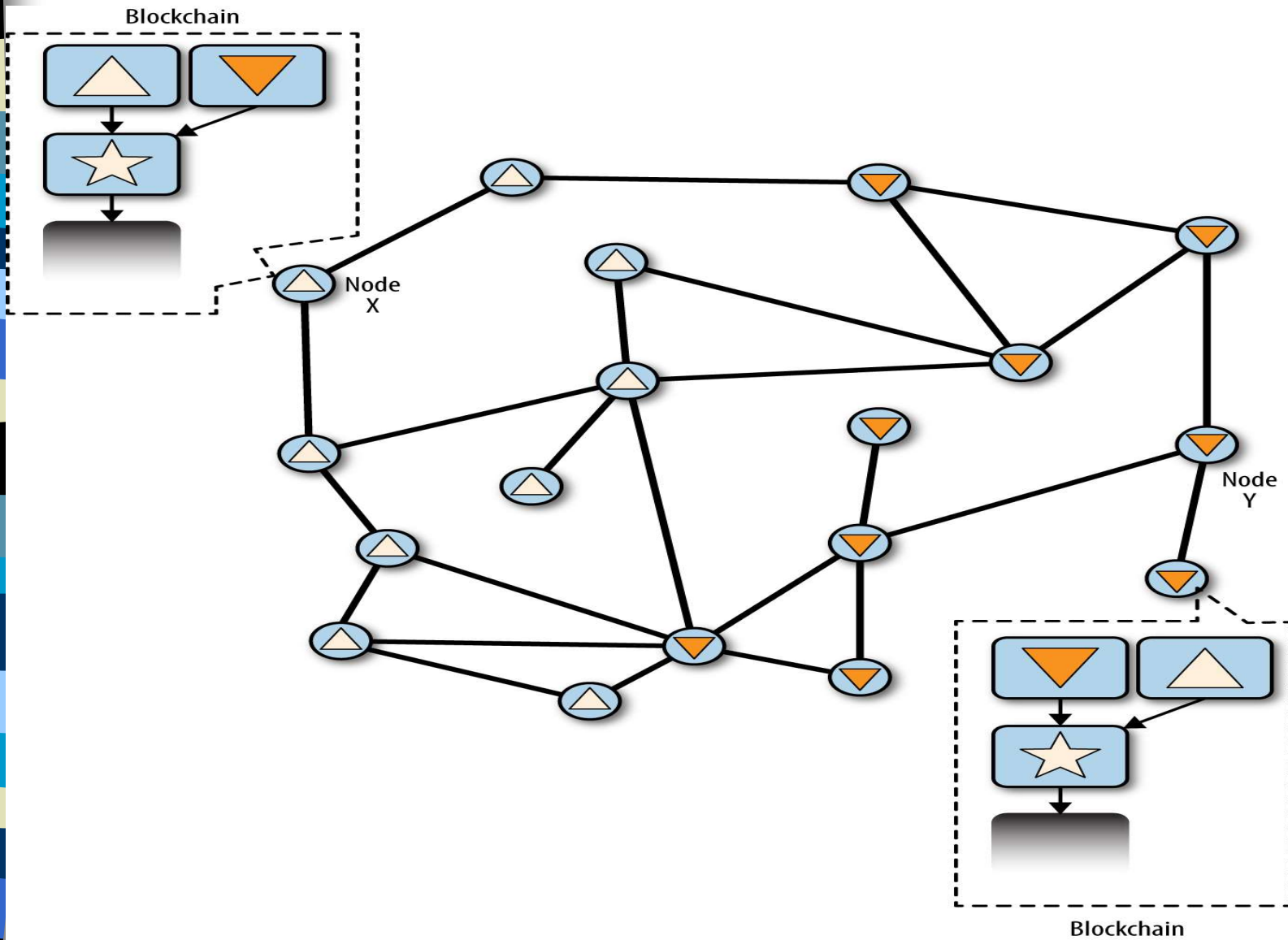
Blockchain assumes a peer-to-peer (P2P) network
No one is in control.



Nodes mine blocks and propagate them locally

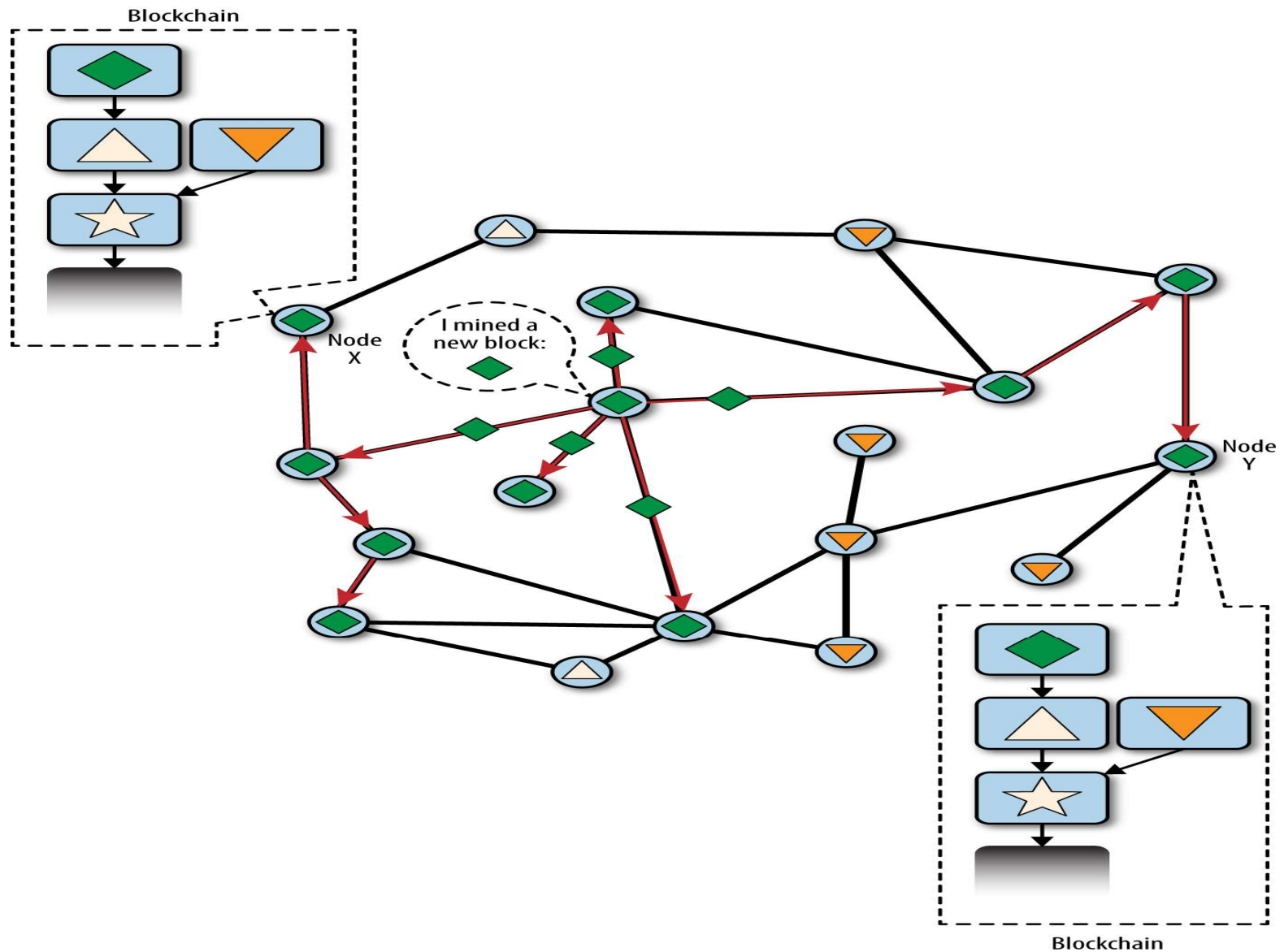


Competing new blocks from different miners

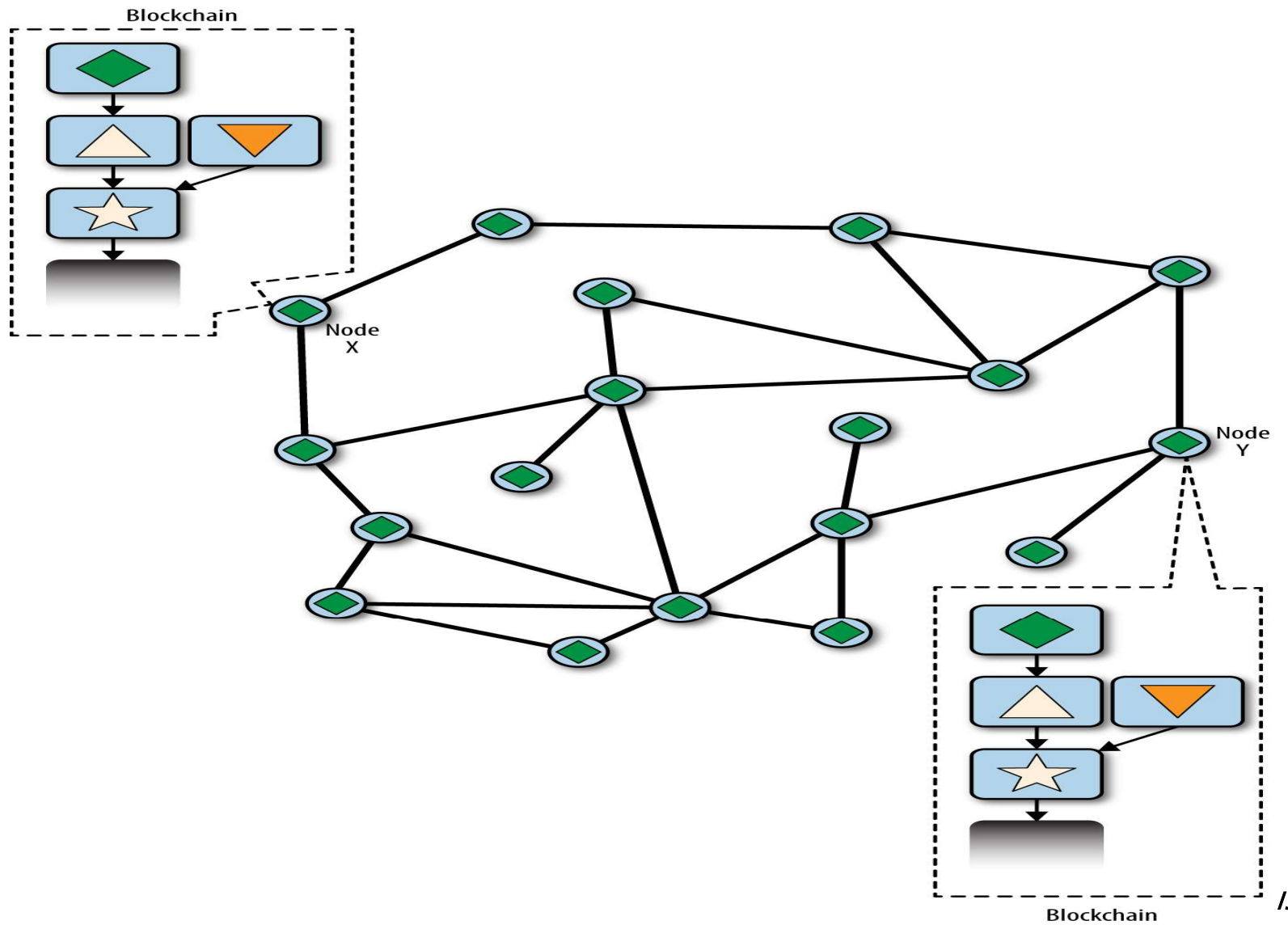


//

Which chain is “longer” (contains more work)?



Eventually consensus is achieved





Resolving differences

- Because of network delays, the longest chain in one part of the network may be different to that in another part of the network.
- Faced with competing blocks, each node adds the different blocks to the existing chain, and then chooses the chain with the greatest cumulative work.
- If the result is a tie, the node waits until another valid block is received.
- If the result is again a tie, the process continues until a clear answer emerges.
- Most often, the conflict (called a temporary fork) is resolved within 2 or 3 blocks (ie, 20-30 minutes)
 - The worst case involved 24 blocks (but due to a bug in a software upgrade).



Summary I

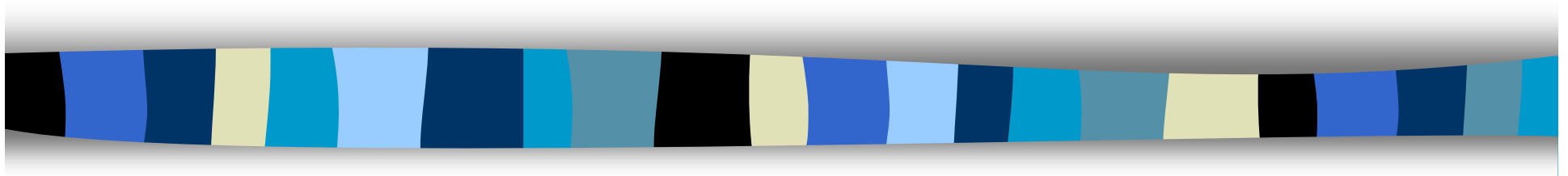
- The Bitcoin Blockchain uses a consensus protocol for all nodes in the network to agree on addition of each new block to the existing blockchain.
- The decisions are made independently and automatically by each node, each one using the exact same algorithm.
- The decision by each node is made on basis of information received from their neighbours (new TXs and new blocks).
- The system works well if all nodes are acting correctly and in good faith.



Summary 2

- The system does not work well if nodes have bugs or are acting maliciously.
- But this should be only temporary
- If a malicious node has inserted a false block this will eventually be superseded by other valid blocks.
- To maintain the false block in the chain over an extended period, the malicious node needs to keep mining new blocks on top of the false block.
 - But this is costly for the malicious node, since it requires the node to do ongoing work in the PoW process.

Thankyou!





Appendix

- The next slides are for your interest only. They are not part of 6CCS3AIN and are not examinable.



Mining new bitcoin

- New bitcoin are created during the creation of each block at a fixed and diminishing rate, approx. every 10 minutes.
- Every 210,000 blocks (ca. four years), the currency issuance rate is decreased by 50%
 - 2009-2012: 50 new bitcoin earned per block
 - November 2012: Reward became 25 new bitcoin per block
 - July 2016: Reward became 12.5 bitcoin per block
 - 8 April 2020: Reward became 6.25 bitcoin at block 630,000
 - ca. 2137: 1 satoshi per block (block 6,720,000) (99% of all BTC)
 - ca. 2140: After 6.93 million blocks a total of almost 2,099,999,997,690,000 satoshis (almost 21 million bitcoin).
- After that, payment to miners will only be via transaction fees.

Reward for mining is new Bitcoin





Mining problem

- Proof-of-Work is designed to create a hurdle to mining
 - Otherwise, nodes would spin-up multiple sock-puppet nodes to win the reward
 - A form of Sybil attack
- The problems get harder over time
 - To ensure that a new block is created about every 10 minutes.
- Problem: Find the hash a specified object with a nonce parameter which is less than sum pre-specified total.
 - Problem designed to be hard to do and easy to check.
 - Can only be solved by trial and error.



Four parts of decentralized consensus

- A Independent verification of each transaction, by every full node
- B Independent aggregation of those transactions into new blocks by mining nodes
together with demonstrated computation through a Proof-of-Work algorithm
- C Independent verification of the new blocks by every node and assembly into a chain
- D Independent selection, by every node, of the chain with the most cumulative computation demonstrated through Proof-of-Work.



A: Independent verification of transactions

Each node checks against the following list of criteria:

- The transaction's syntax and data structure is correct.
- Neither lists of inputs or outputs are empty.
- The transaction size in bytes is less than MAX_BLOCK_SIZE.
- Each output value, as well as the total, is within the allowed range of values
- None of the inputs have hash=0, N=-1 (coinbase transactions should not be relayed)
- nLocktime is equal to INT_MAX, or nLocktime and nSequence values are satisfied according to MedianTimePast.
- The transaction size in bytes is greater than or equal to 100.
- The number of signature operations (SIGOPS) contained in the transaction is less than the signature operation limit.
- The unlocking script can only push numbers on the stack, and the locking script must match isStandard forms.
- A matching transaction in the pool, or in a block in the main branch, must exist.
- For each input, if the referenced output exists in any other transaction in the pool, the transaction is rejected.
- For each input, look in the main branch and the transaction pool to find the referenced output transaction. If the output transaction is missing for any input, this will be an orphan transaction. Add to the orphan transactions pool, if a matching transaction is not already in the pool.
- For each input, if the referenced output transaction is a coinbase output, it must have at least COINBASE_MATURITY confirmations.
- For each input, the referenced output must exist and cannot already be spent.
- Using the referenced output transactions to get input values, check that each input value, as well as the sum, are in the allowed range of values (less than 21m coins, more than 0).
- Reject if the sum of input values is less than sum of output values.
- Reject if transaction fee would be too low (minRelayTxFee) to get into an empty block.
- The unlocking scripts for each input must validate against the corresponding output locking scripts.



Four parts of decentralized consensus: C & D

- C Independent verification of the new blocks by every node and assembly into a chain
- D Independent selection, by every node, of the chain with the most cumulative computation demonstrated through Proof-of-Work.
 - We can reference blocks by their height (about 660,000), or the hash of their header.
 - Block height may not be unique (if there is a fork).
 - Block hash is not stored within the block
 - It is calculated by each node as the block is received.



Validating a new block

Criteria for validation include:

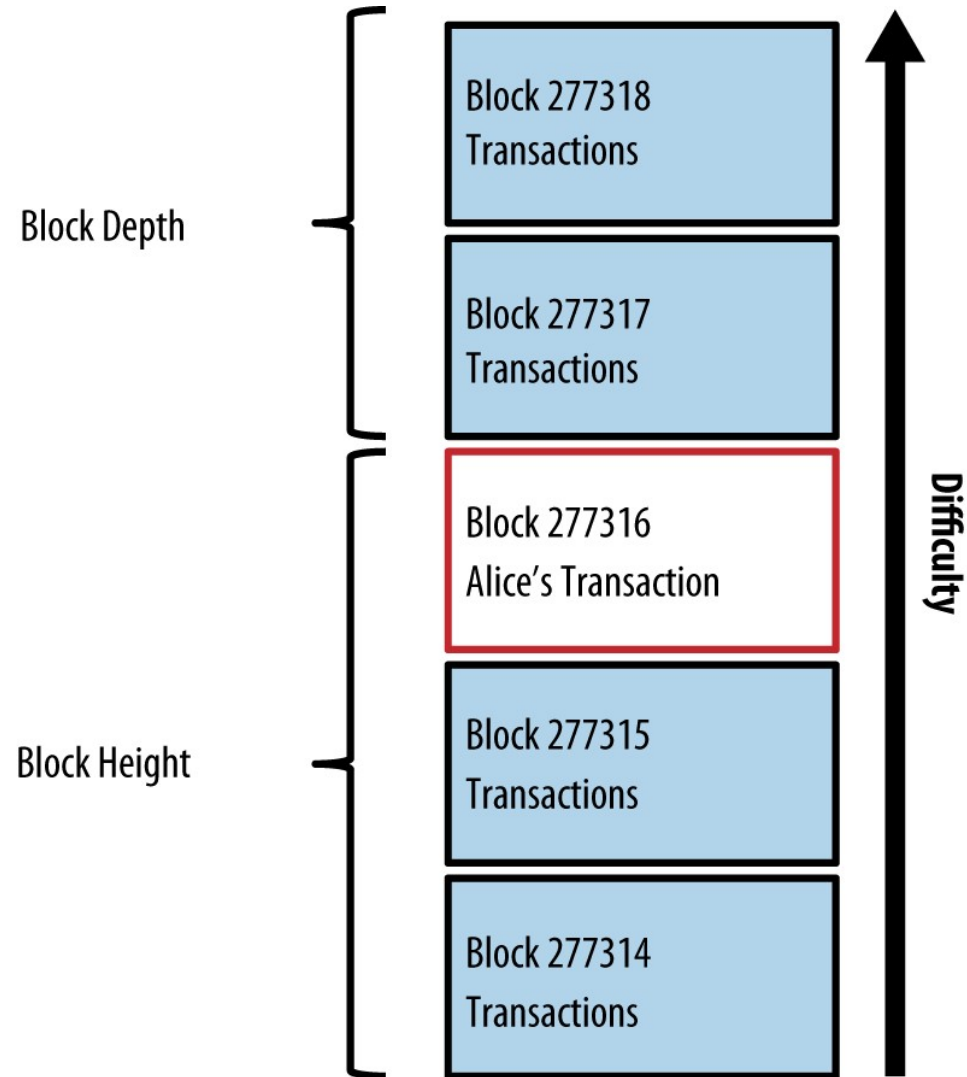
- The block data structure is syntactically valid
- The block header hash is less than the target (enforces Proof-of-Work)
- The block timestamp is less than two hours in the future (allowing for time errors)
- The block size is within acceptable limits
- The first transaction (and only the first) is a coinbase transaction
- All transactions within the block are valid using the transaction checklist for Independent Verification of Transactions.



How do nodes decide between competing blocks?

- Nodes keep three collections of blocks
 - Those on the main blockchain
 - Those that form branches off the main blockchain
 - Orphan blocks – those without a parent block
- The main chain is the chain with the most cumulative difficulty associated with it
 - Usually the chain with the most blocks
 - If two chains are equal length, then the main chain is the one with most PoW
 - Forks are usually resolved within 1-3 blocks
- 10 minutes for each block time is a compromise between
 - Fast confirmation times & the probability of a fork.
- As computer power increases, the Bitcoin blockchain automatically adjusts other parameters to ensure average block time is around 10 minutes.

Block height currently is about 659,050 (on 2020-11-28)



<https://blockchain.info/q/getblockcount>