Explanation of *sdbft* consensus algorithm.

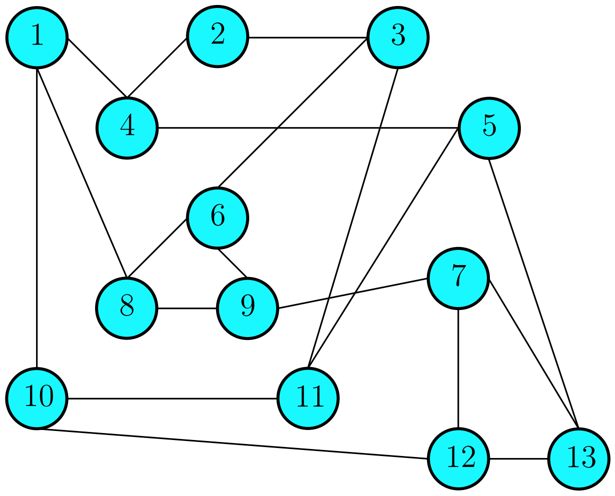
rr@sumus.team, a@sumus.team

August 26, 2019

1.1.1 Explanation of ***sdbft*** consensus algorithm

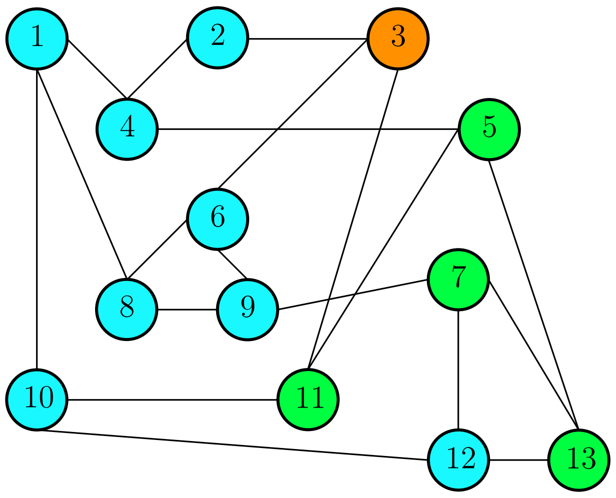
This section contains a simplified description of the ***sdbft*** consensus algorithm.

1. Before work starting, we have a peer-to-peer network with the nodes that have their own network address and a unique number that all network participants know. For example, we will have 13 network participants. Let's number all nodes with the following numbers: 1 - 13. We also agree that 5 nodes will make the consensus.

   The blue color will indicate the nodes working in the block chain. Green will indicate the nodes participating in the consensus. The master node will be marked in orange. The blocks in abnormal operation mode will be marked by red.

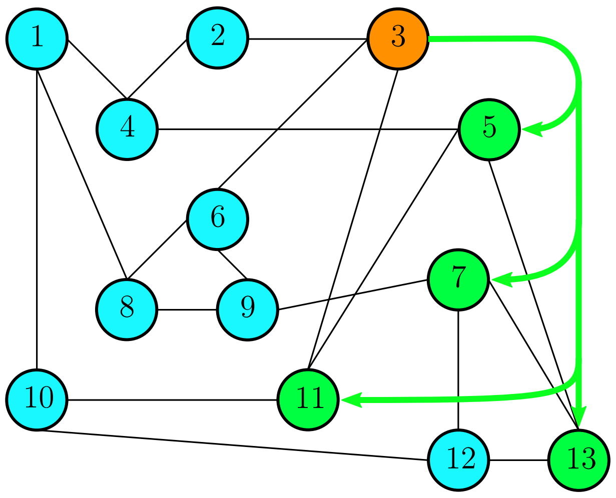
2. Consensus operation start. Let all nodes of the peer-to-peer network accept the block 1. The accepted block contains information that will allow the function f (see Appendix A) to create a random sequence. Let this sequence be the following — No. 3, 5, 7, 11, 13. Let's mark the nodes — No. 3, 5, 7, 11, 13 with color.



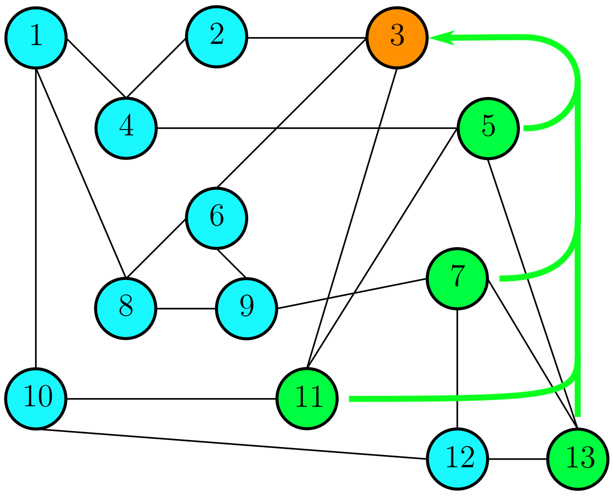
3. According to the figure above, the node 3 is marked in orange to indicate that it is a master node. From this point on, the nodes 3, 5, 7, 11, 13 participate in consensus.

4. Let the node 3 has a new transaction from the node 2. The node 3 checks whether the transaction is correct. If it is recognized as correct, then node 3 sends it to the nodes 5, 7, 11, 13.

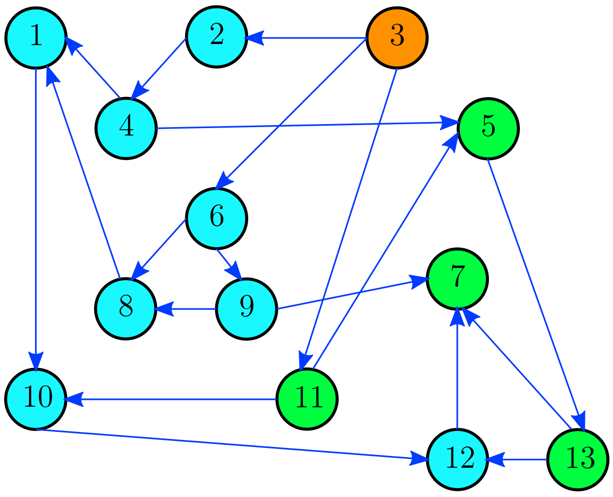
5. At the end of the time specified for block closing, node 3 sends the nodes No. 5, 7, 11 and 13 a message on block closing.



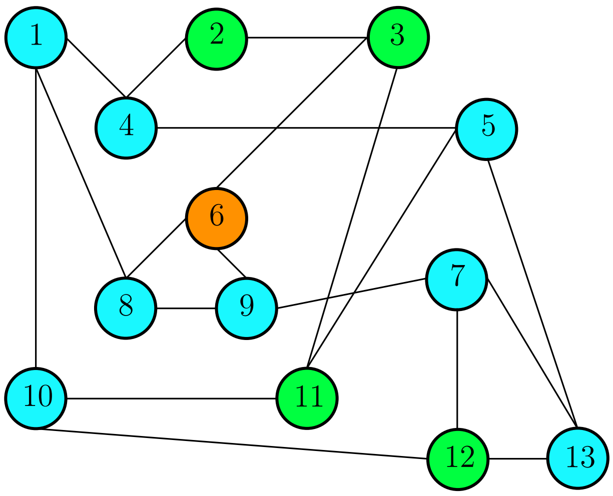
6. The nodes 5, 7, 11 and 13 forward the hash of the Merkle tree, the transactions they received, and their signatures under the hash to the node 3.



7. Node 3 considers signatures, if the signatures are correct and their number satisfies the solution of the Byzantine generals problem and if there are at least 3 of them, then the block is considered as formed. The node 3 sends out a new block announcement to all network nodes.



8. The nodes accept the block number 2. We return to the second step of the algorithm, now let the function f (see Appendix A) create a random sequence based on the received blocks, let there be the numbers 6, 2, 3, 11, 12.



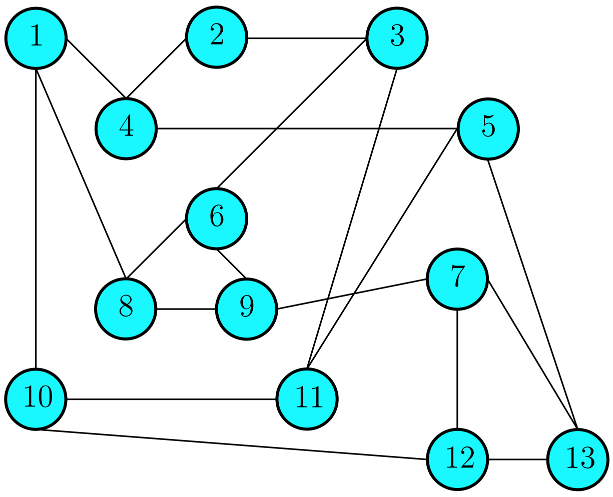
Next, the process is repeated in accordance with the step 3 - 8 of the algorithm.

1.1.2 Error handling by *sdbft* algorithm

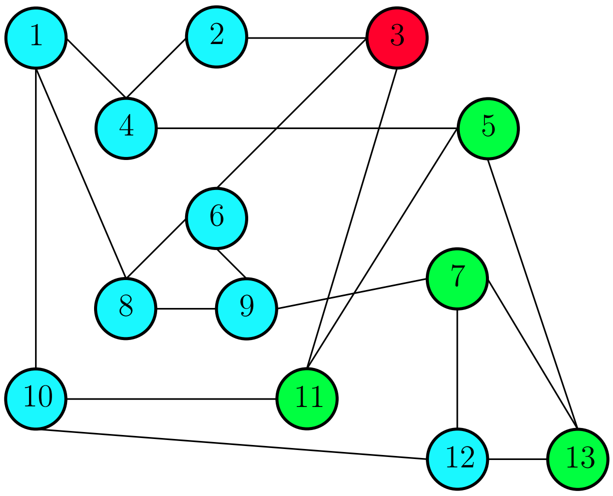
Master node unavailable

1. Let's repeat the steps of the algorithm 1 and 2.

2. Before work start, we have a peer-to-peer network with the nodes that have their own network address and a unique number that all network participants know. For example, we will have 13 network participants. Let's number all nodes with the following numbers: 1 - 13. We also agree that 5 nodes will make the consensus.

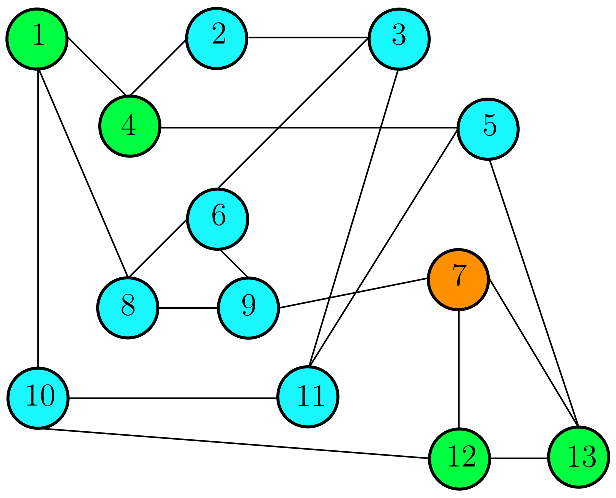


3. Consensus operation start. Let all nodes of the peer-to-peer network accept the block 1. The accepted block contains information that will allow the function f (see Appendix A) to create a random sequence. Let this sequence be the following - No. 3, 5, 7, 11, 13. The node 3 is unavailable. Let's mark the network nodes with color.



4. Escort nodes will not receive a message about block closure, the block chain network will go to the next round (see Appendix A).

5. Let's return to the second step of the algorithm, now let the function f (see Appendix A) create a random sequence based on the received block and round number, let there be the numbers 7, 1, 4, 12, 13. The block chain network will look as follows.

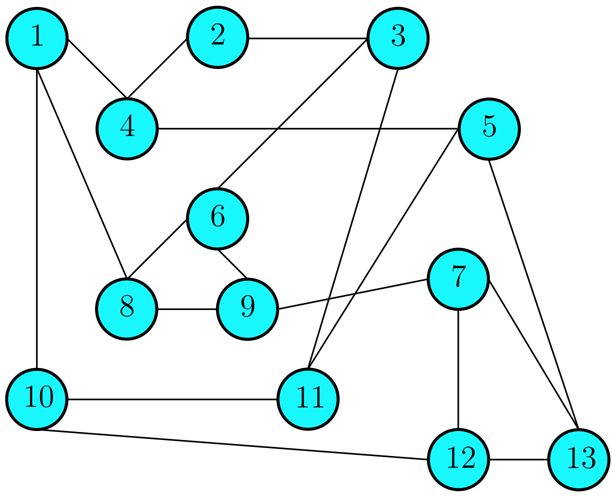


6. Next, the algorithm will be executed in a regular manner according to the steps 3 - 8.

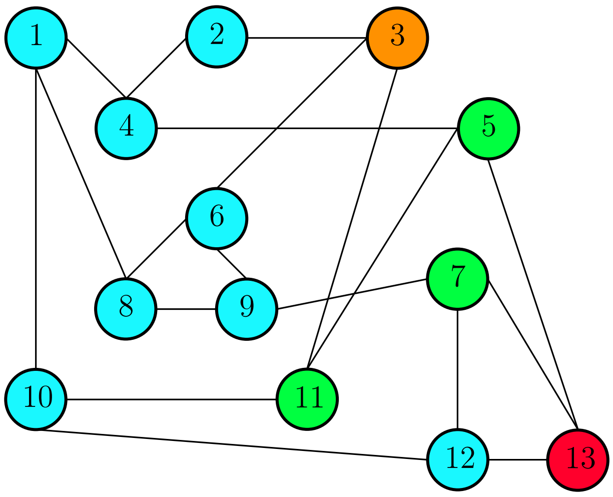
Escort node unavailable

1. Let's repeat the steps of the algorithm 1 and 2.

2. Before work start, we have a peer-to-peer network with the nodes that have their own network address and a unique number that all network participants know. For example, we will have 13 network participants. Let's number all nodes with the following numbers: 1 - 13. We also agree that 5 nodes will be the part of the consensus.



3. Consensus operation start. Let all nodes of the peer-to-peer network accept the block No. 1. The accepted block contains information that will allow the function f (see Appendix A) to create a random sequence. Let this sequence be the following one - 3, 5, 7, 11, 13. The node 13 is unavailable. Let's mark the selected nodes with color.



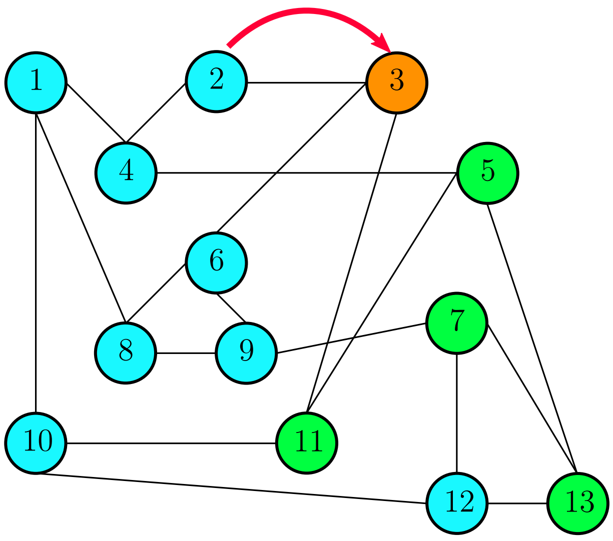
4. Escort node 13 will not receive a message about block closure and will not send its signature to close the block. If the remaining three nodes send the correct transaction signatures of the generated block, then the block will be formed. The nodes participating in the consensus will be selected again.

5. If the remaining three nodes send incorrect transaction signatures of the generated block, the block will not be generated. The block chain will move on to the next round. The nodes participating in the consensus will be selected again.

Invalid transaction

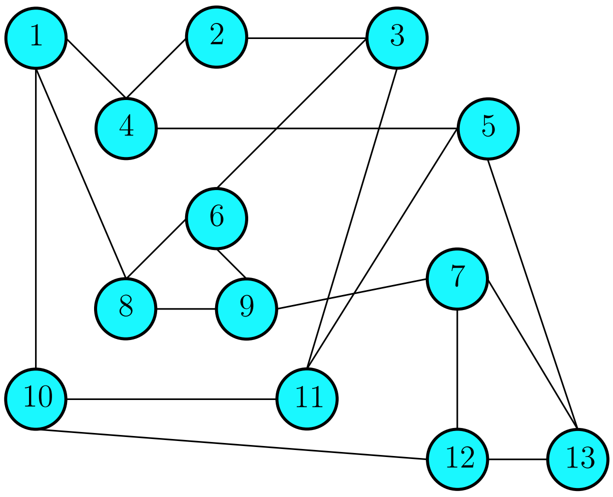
1. Let the node 3 received a new transaction from the node 2. The node 3 checks the transaction and recognizes it as incorrect.

2. If the node 3 recognized the transaction as invalid, then it discards it. A message for the node 2 on the incorrect transaction is not forwarded to the nodes included in the consensus.

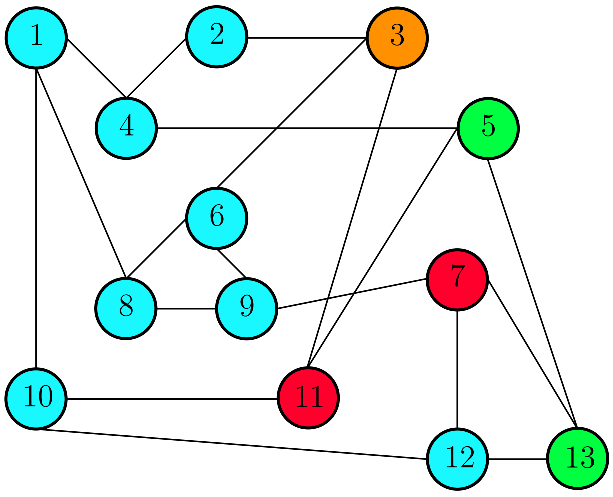


Different block chain number of blocks on different nodes
A different number of accepted blocks on different nodes of the block chain may be different if, for example, the network was segmented and not all nodes managed to synchronize. Let's repeat the steps 1 - 5 of the algorithm.

1. Before work start, we have a peer-to-peer network with the nodes that have their own network address and a unique number that all network participants know. For example, we will have 13 network participants. Let's number all nodes with the following numbers: 1 - 13. We also agree that 5 nodes will make the consensus.

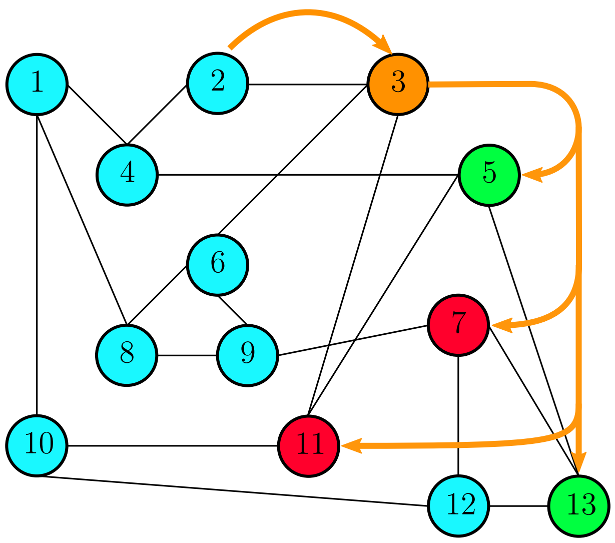


2. The beginning of the consensus operation. Let all nodes of the peer-to-peer network accept block 1. The accepted block contains information that will allow the function f (see Appendix A) to create a random sequence. Let this sequence be the following — 3, 5, 7, 11, 13. The nodes 7 and 11 have an excellent number of received blocks from the nodes 3, 5, 13. Let's mark the selected nodes with color.

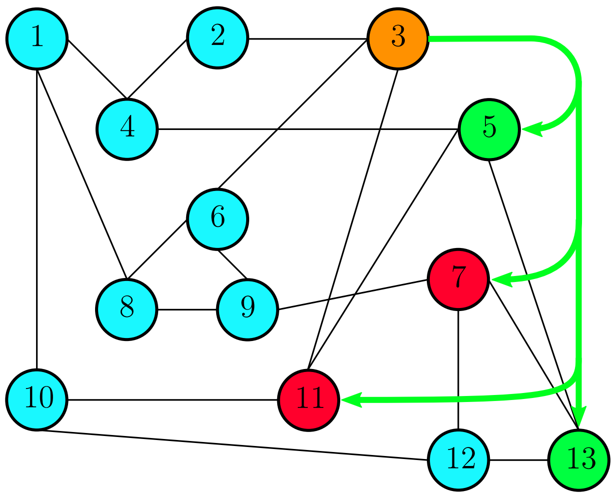


3. As is shown by the figure, the node 3 was marked in orange to show that it is a master node. From now on, the nodes 3, 5, 13 participate in consensus.

4. Let the node 3 received a new transaction from the node 2. The node 3 checks if the transaction is correct. If it is recognized as correct, then the node 3 sends it to the nodes 5, 7, 11, 13. The nodes 7 and 11 reject the transaction.

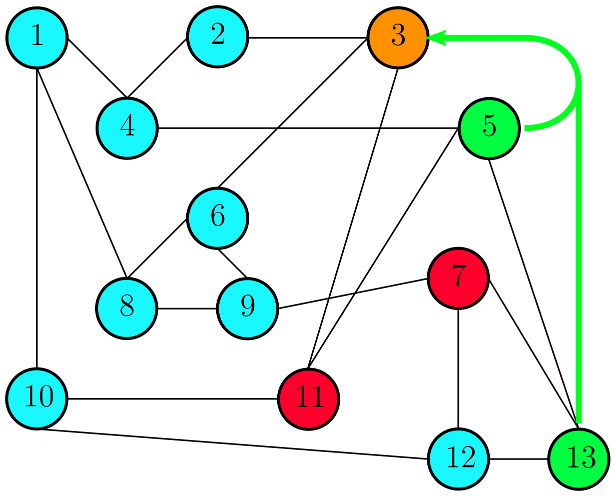


5. At the end of the time for the block closure, the node 3 sends a message about the block closing to the nodes 5, 7, 11, 13. The nodes 7 and 11 reject the message.



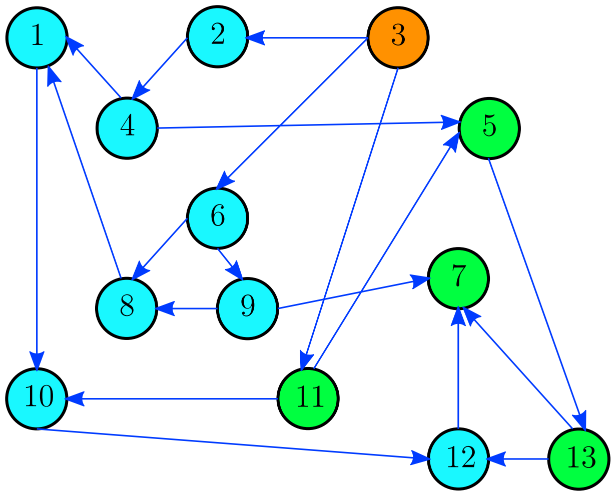
6. The nodes 5 and 13 forward the hash of the transactions and their signatures under the hash to the node 3.

7. The node 3 checks the signatures of escort nodes. Since the number of escort node signatures is not enough to accept a block, the block chain moves on to the next round.

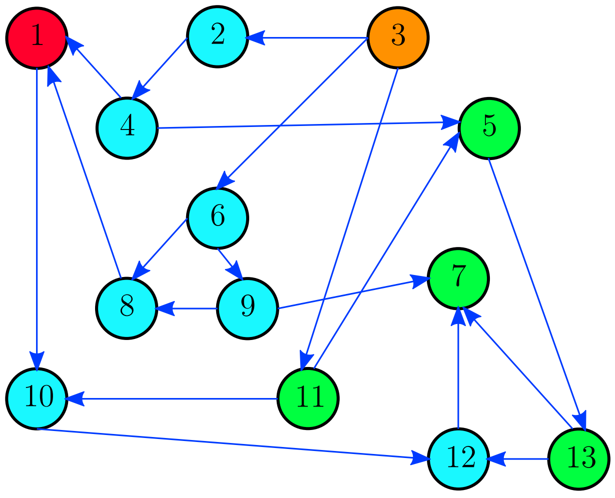


New blockchain block is rejected by node
A network node can reject a new blockchain block. There may be several reasons why the node rejects the block. For example, a database reading error occurred on the node due to a software or hardware failure and wallet balances changed. Let's repeat the step 7.

1. The node 3 sends out a new block announcement to all network nodes.



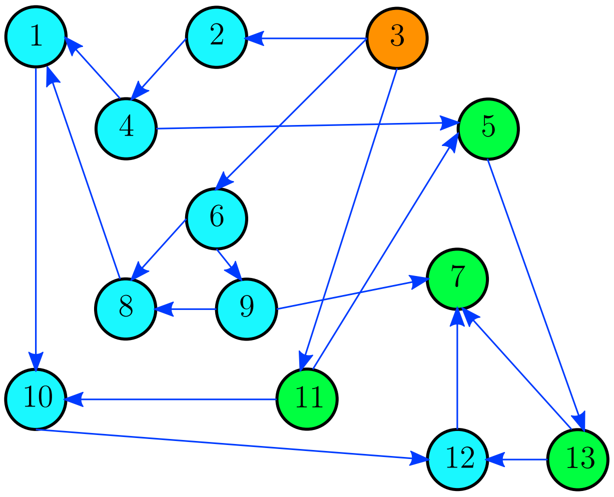
2. Let the node number 1 rejects the block.



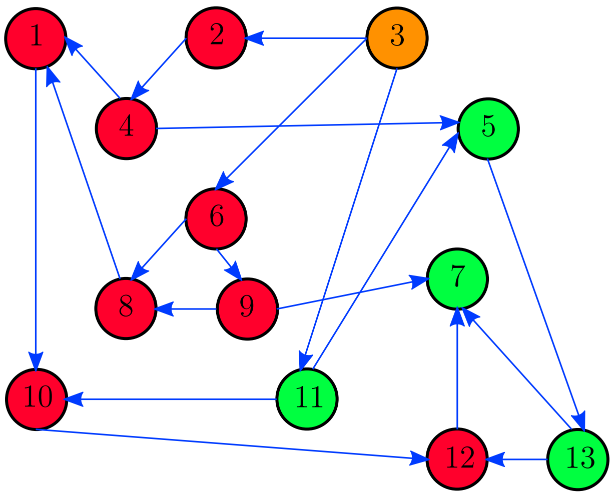
3. The node 1 tries to find a block on the network with a hash sum that is different from the block it recognized as an error; if the node cannot find a block satisfying it, the node starts the block chain re-synchronization procedure, see step 3.5.

Network rejects a new blockchain block
When a new block is created, a conscious attempt of a group of nodes to impose its own erroneous block may occur theoretically. Let the nodes No. 3, 5, 7, 11, 13 try to impose their own, the wrong block of the block chain network. Let's repeat the step 7.

1. The node 3 sends out a new block announcement to all network nodes.



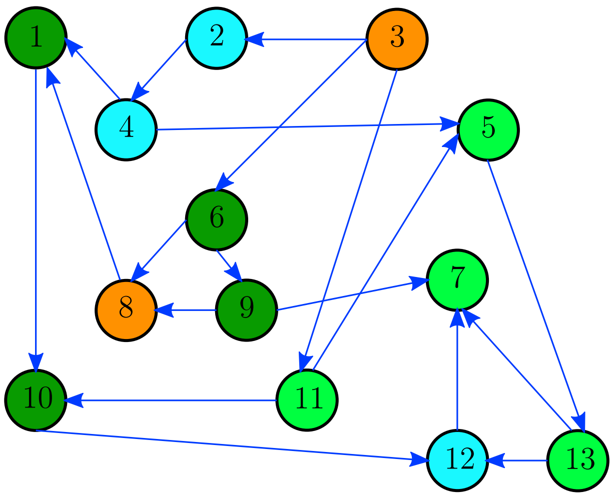
2. All network nodes reject a new block.



3. The block chain moves to the next round until the next block is correctly formed.

Two blocks with identical numbers appeared in the network
Two blocks with identical numbers can occur in the network only if two consensuses are formed from the nodes with different rounds, and this is possible if a global failure occurs in the network. For example, some nodes were located on the servers that were simultaneously rebooted. In this case, the following will occur.

1. Suppose that two sets of nodes have been formed to create consensus - 3, 5, 7, 11, 13 and 8, 9, 6, 1, 10. The node 3 and 8 send out a new block announcement to all network nodes.



2. When a new block is accepted, the node checks whether the block creation round is included in the confidence interval of the rounds or not. I.e. how much a new block round differs from its own node round. If the round is recognized by the node as correct, then the block is accepted. If it is considered incorrect, the block is rejected. Further, there are two possible ways of the situation development:

   a. The rounds of the formed blocks are in the confidence interval. In this case the node will receive the block that arrived first. The probability of getting into a block chain block will depend on the block accepted by most network nodes.

   b. The round of one of the formed blocks is not in the confidence interval for the majority of the network nodes. Therefore, a block with a round number from the confidence interval will be accepted by most nodes.