Chapter 2
Toward Smart Contract and Consensus Mechanisms of Blockchain

This chapter presents a detailed description of smart contract and proof-based consensus mechanisms used in blockchain technology. Smart contracts are used in blockchain technology while making transaction between two parties, and it enables only the validated transaction to be included in the blockchain. However, to validate a transaction, Satoshi Nakamoto, who is the inventor of blockchain, introduced a proof-of-work (PoW) consensus algorithm while performing transaction among blockchain nodes (i.e., users) [1]. Later on, various proof-based consensus mechanisms [e.g., proof of stake (PoS), proof of location (PoL), PBFT (practical Byzantine fault tolerance)] are proposed. The main concept of applying proof-based algorithms is that the nodes within the blockchain network that performs and exhibits sufficient proof will get the privilege to append a new block to the main chain and collect the reward. In this chapter, we present a clear explanation of smart contract and discuss about some of the most important consensus algorithms of blockchain that are widely used (Fig. 2.1).

2.1 Smart Contract

A smart contract is an important part that contributed to the advancement of blockchain technology (Fig. 2.2). Smart contracts were introduced in the 1990s as a computer-based transaction protocol that can document, control, and execute legal events and perform actions based on the contract agreement [2]. The main objectives of integrating smart contracts in blockchain technology are to reduce the need for trusted intermediators and fraud losses and to reduce the malicious, enforcement costs, and accidental exceptions [3]. A smart contract is implemented on top of the blockchain technology. In a smart contract, when two parties agreed on all the clauses stated in a smart contract and start the transaction, it is automatically embedded into the blockchain and proceeded when certain clauses or conditions are

A. Imteaj et al., *Foundations of Blockchain*, SpringerBriefs in Computer Science,
https://doi.org/10.1007/978-3-030-75025-1_2

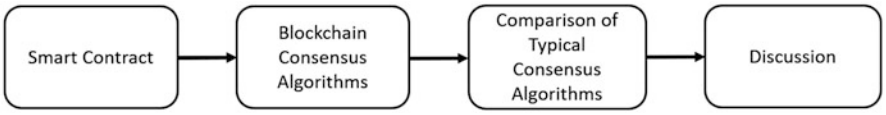
Smart Contract
Blockchain Consensus Algorithms
Comparison of Typical Consensus Algorithms
Discussion

Fig. 2.1 Chapter overview

fulfilled (e.g., a party gets penalty or punishment when it breaches the smart contract clauses). The contractual clauses of a smart contract are transformed into executable computer programs. The logical connections that exist among the contractual clauses are preserved in the form of program logic flows (e.g., if-else-if statements) [4]. A contract statement that is executed cannot be further modified, i.e., the transaction is recorded as immutable in the blockchain. Besides, smart contracts facilitate access control of each smart contract function. That means it is possible to set access permission for different functions of smart contracts. Moreover, smart contracts ensure contract enforcement, i.e., it guarantees that the execution of smart contract is deterministic. In particular, whenever any smart contract conditions are satisfied, it automatically triggers the corresponding function. For instance, two users of a blockchain network named Alice and Bob give their consent on the penalty or punishment of violating a smart contract agreement. If Alice breaches the contract, then immediately, the penalty mentioned in the smart contract will be cut from Alice's deposit. The life cycle of a smart contract has four phases: creation, execution, deployment, and finalization or completion [4, 5], which is illustrated in Fig. 2.3.

2.1.1 Creation of a Smart Contract

At the beginning of a contract creation, the involved parties perform negotiation on the rights, obligations, and prohibitions on that contract. All the involved parties reach to an agreement after multiple rounds of negotiations and discussions. A draft of an initial contractual agreement is prepared by the lawyers or counselors of the involved parties. After that, the agreement written in natural languages is converted into a programmable language that may include declarative and logic-based rule language [5, 6]. The conversion of a smart contract from a natural language to a computer-based language is composed of several stages, i.e., contract design, deployment, testing, and validation, like software development. After several iterative processes, a smart contract is prepared and published in the blockchain.

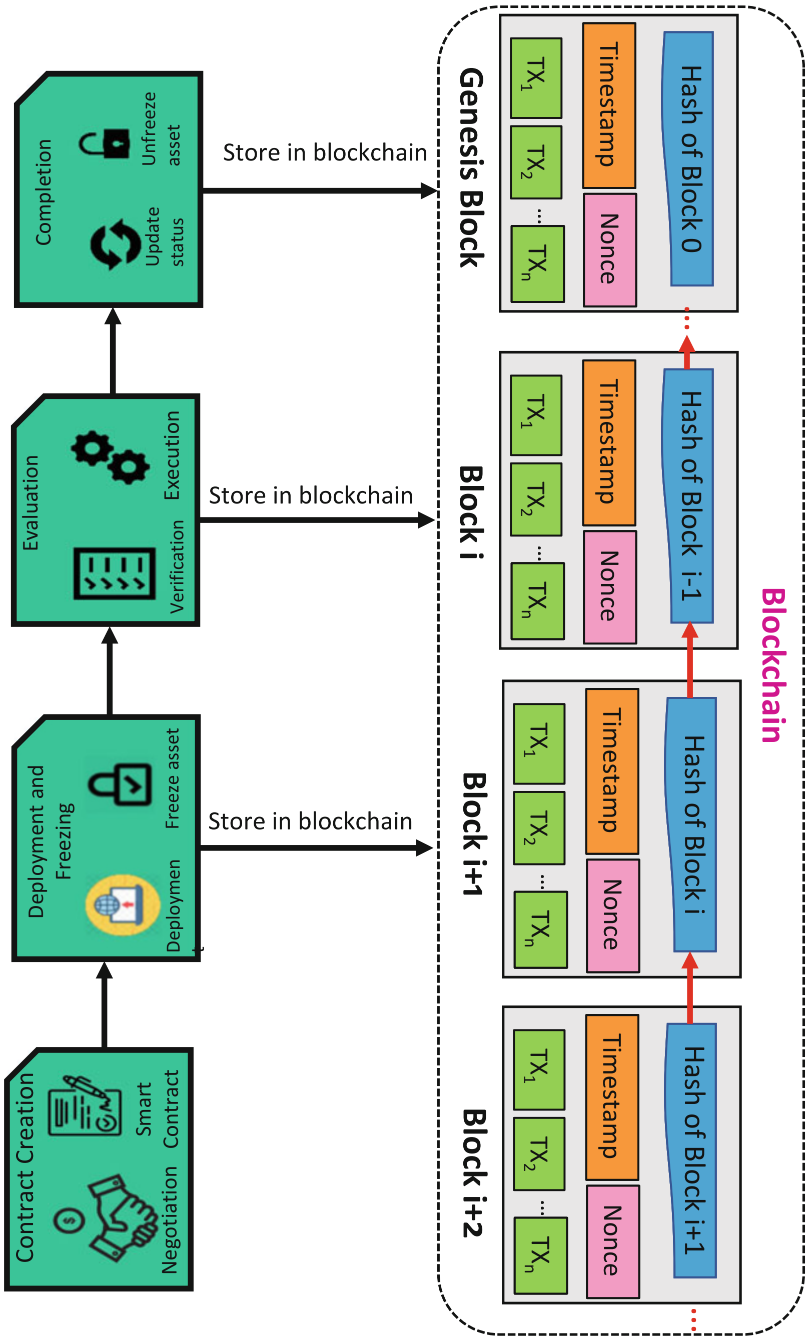

Fig. 2.2 Steps of preparing a smart contract

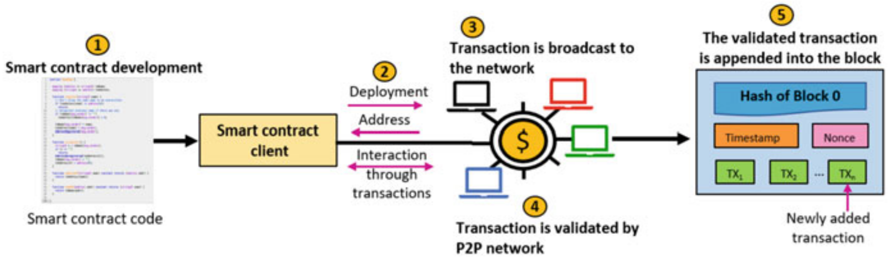
1
Smart contract development
Smart contract code
Smart contract client
2
Deployment
Address
Interaction through transactions
3
Transaction is broadcast to the network
$
4
Transaction is validated by P2P network
5
The validated transaction is appended into the block
Hash of Block 0
Timestamp
Nonce
TX1
TX2
...
TXn
Newly added transaction

Fig. 2.3 Life cycle of a blockchain smart contract

2.1.2 Deployment of a Smart Contract

After the smart contract is prepared, it is validated and deployed on top of a blockchain. After a contract is published on a blockchain, it cannot be modified due to blockchain immutability. Any correction or modification requires creating a new contract. When a contract is published and implemented on the blockchain, it is accessible to all the available parties. In the meantime, the involved parties' digital assets are frozen via their digital wallets, and those digital wallets are sued to identify the parties [7].

2.1.3 Execution of a Smart Contract

After the smart contract is deployed in blockchain, the contractual clauses are monitored and evaluated. Whenever a contractual condition meets (e.g., receive the contractual product), the sequential related procedures are automatically executed. It should be noted that a smart contract consists of a couple of declarative statements that may have logical connections. Therefore, once a condition is triggered, the connected statements are automatically executed and validated by blockchain miners. The verification and execution contract status is written on the blockchain.

2.1.4 Completion of a Smart Contract

After the evaluation of a smart contract, the status of the involved parties is updated. All the executed transactions and the updated states of all parties are written in the blockchain. Meanwhile, the involved parties transfer digital assets according to the contract policy. After the completion of digital asset transfer, the asset is unlocked, and it completed the life cycle of a smart contract.

It is to be noted that during the deployment and freezing, evaluation, and completion stage of a smart contract life cycle, a sequence of transactions is executed, written in the blockchain (see Fig. 2.3).

2.2 Blockchain Consensus Algorithms

If every node broadcasts blocks that hold verified transactions in a blockchain network, then confusion could arise. For instance, if we consider a verified transaction that needs to be inserted into the block and multiple nodes perform broadcasting about that, we can observe redundant transactions in different blocks. Thus, the ledger would be meaningless; rather than, a node needs to be selected that will be responsible for inserting the transaction into a block. After successfully inserting all the verified transactions into the block, the nodes within a blockchain network need to verify and reach an agreement (called *consensus*) to insert a new block to the main chain. Each participated node needs to prove its validity and authenticity before inserting a new block to the chain. The first node that is able to accomplish consensus will have the right to insert the new block. Once a block is verified and inserted into the chain, it is infeasible to modify or delete that block [8, 9]. We can achieve consensus in a blockchain network by applying various proof-based algorithm. In this section, we discuss various consensus models and how they actually work.

2.2.1 Proof-of-Work (PoW) Consensus Model

The most popular and widely used consensus mechanism is proof of work (PoW), which is used in Bitcoin. In currency cryptography, PoW has been a popular choice for many years. Under PoW, a participant is called a miner. The miners perform a computational task, i.e., solve a puzzle problem in order to obtain the right to generate a new block. The nodes perform puzzle solving by finding a specific hash function. Each blockchain miner tries to solve the hash value, i.e., they try to figure out a particular value as a nonce to meet a predefined hash condition, e.g., finding the nonce value that will make the first 30 bits of its hash to zero. The difficulty of the consensus mechanism can be increased by setting a hard puzzle. Each miner tries to find hash values that are smaller than or equal to a certain target value to reach consensus in the blockchain network [10, 11]. Whenever a miner finds the target hash value, it broadcasts the current block to the whole network. After that, all other nodes verify the correctness of its hash value. If the block is legit, all nodes append that new validated block to their blockchain. In Fig. 2.4, we presented an illustration of proof-of-work (PoW) consensus mechanism.

The main advantage of PoW is its strong security, an acceptable level of scalability, and decentralization feature. However, the block mining and validation

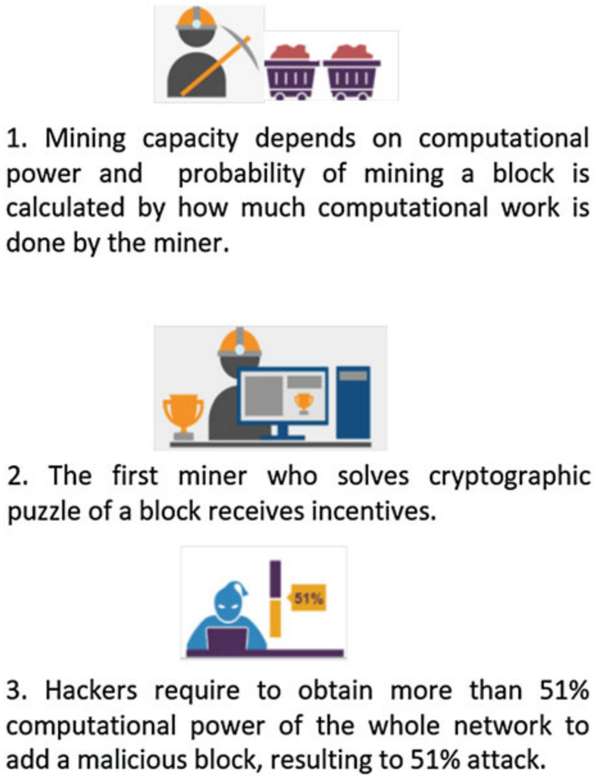

Fig. 2.4 Proof-of-work (PoW) consensus mechanism

1. Mining capacity depends on computational power and probability of mining a block is calculated by how much computational work is done by the miner.

2. The first miner who solves cryptographic puzzle of a block receives incentives.

3. Hackers require to obtain more than 51% computational power of the whole network to add a malicious block, resulting to 51% attack.

process of PoW wastes a lot of energy. As more difficult the puzzle is, the more computational power is required. Therefore, the resource-constrained nodes would not solve a complex puzzle as the target hash generation time and success rate depend on the node's computational ability [12]. In summary, the disadvantages of PoW are high block creation time, high energy requirement, less throughput, and special hardware dependency.

2.2.2 *Proof of Stake (PoS)*

The next common and widely used distributed consensus algorithm is proof of stake (PoS). The main aim of creating PoS concept is to eliminate the main drawback of PoW, i.e., energy inefficiency. In PoS-based approach, the creator of next block is selected via different combinations of random selection, stake supply, and age that could provide scalability. The idea of PoS was first introduced in 2011, particularly for Peercoin cryptocurrency, and later on was used in others, i.e., Nxt and Blackcoin [13]. In this consensus mechanism, the node for creating next block is selected via a quasi-random process, and the selection is performed by considering the assets stored in the wallets of the nodes. As this method does not require high computational power to validate any proof, the miners do not receive any reward without the transaction fees. The advantages of PoS are energy efficiency, high throughput, fast block creation time, scalability (less than PoW), and no dependence or requirement of special hardware of the nodes. Although PoS does not require computation power like PoW, it depends on the nodes that possess the most stake, and eventually, blockchain may become centralized. Another common problem of PoS is called "nothing at stake"; it means that if a node within a blockchain network has nothing in its stake while misbehaving, then the node would not be afraid of losing anything. Therefore, the node will not face any obstacle that prevents it from misbehaving within the network. One example of node misbehavior could be constructing two sets of new blocks to get double transaction fees [14, 15]. In Fig. 2.5, we presented an illustration of proof-of-stake (PoS) consensus mechanism.

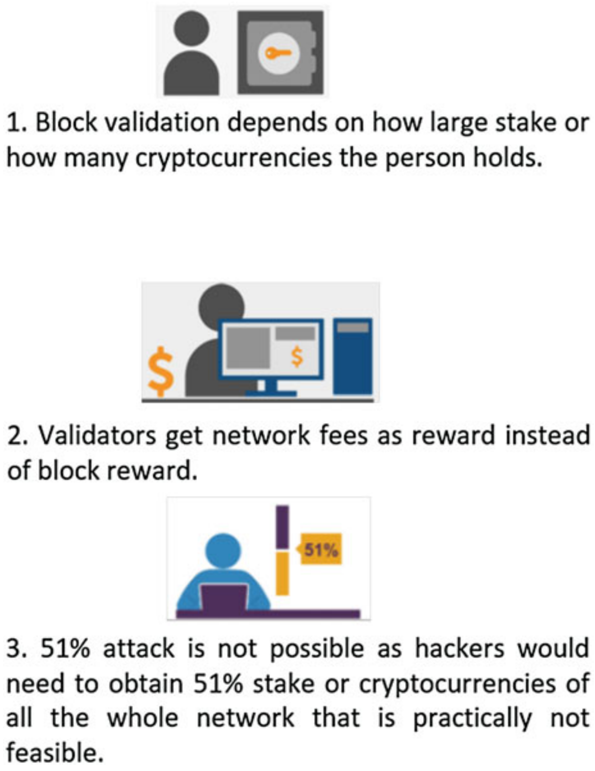
Fig. 2.5 Proof-of-stake (PoS) consensus mechanism

1. Block validation depends on how large stake or how many cryptocurrencies the person holds.

2. Validators get network fees as reward instead of block reward.

3. 51% attack is not possible as hackers would need to obtain 51% stake or cryptocurrencies of all the whole network that is practically not feasible.

2.2.3 Practical Byzantine Fault Tolerance

Practical Byzantine fault tolerance (PBFT) is a replication algorithm that can tolerate Byzantine faults [16], inconsistency issues due to unreliable nodes within the system, and can perform efficiently than previous approaches [16, 17]. Today's malicious attacks on software can be an outcome of arbitrary or Byzantine behavior of faulty nodes. PBFT consensus algorithm is a formation of responsive machine code and used for solving such Byzantine general problems. In the PBFT method, all blockchain nodes must participate in voting to include the next block to the chain. The consensus is achieved if more than two-thirds of the blockchain nodes agree that the block and PBFT method withstand the remaining one-third nodes' opinion. Otherwise, consensus will not be achieved. In this way, PBFT achieves faster consensus and is comparatively more economical than PoW. Besides, the PBFT does not require any asset in the stake of the nodes for the consensus process [15]. In summary, PBFT has energy efficiency and high throughput that can be considered as its advantages. There remain possible delays as the blockchain network may need to wait to receive votes from all nodes, and the security scheme is not stronger than PoW. In Fig. 2.6, we presented an illustration of practical Byzantine fault tolerance (PBFT) consensus mechanism.

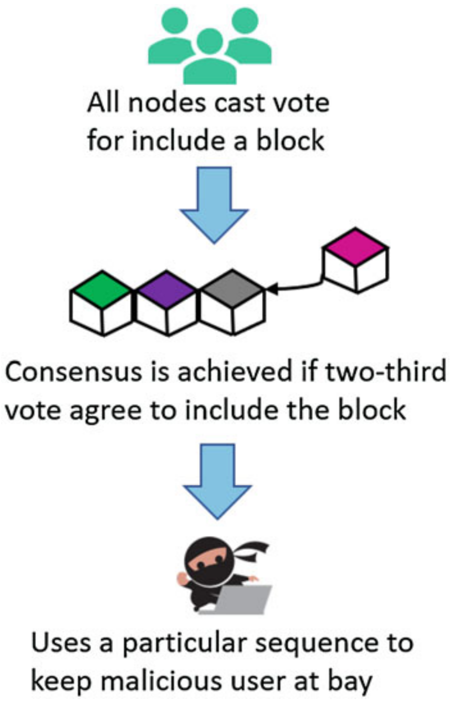
All nodes cast vote for include a block
Consensus is achieved if two-third vote agree to include the block
Uses a particular sequence to keep malicious user at bay

Fig. 2.6 Practical Byzantine fault tolerance (PBFT) consensus mechanism

2.2.4 Proof of Elapsed Time

Proof of elapsed time (PoET) is another consensus mechanism that uses a fair lottery protocol. Through fair lottery protocol, PoET consensus scheme prevents high energy consumption and high resource utilization by the blockchain nodes. This consensus algorithm uses a random elapsed time to choose nodes that are going to obtain mining rights and select block winners. Besides, PoET consensus mechanism improves transparency by certifying that lottery results of PoET protocol are verifiable by outsider or external participants. In summary, the workflow of PBFT is similar to PoW consensus algorithm without high power consumption of nodes. PBFT ensures power efficiency by allowing a blockchain miner to sleep and the nodes to switch to other tasks as per their convenience. In Fig. 2.7, we presented an illustration of proof-of-elapsed-time (PoET) consensus mechanism.

2.2.5 Proof of Activity

Another consensus algorithm is proof of activity (PoA), which is a hybrid consensus method that integrates both PoW and PoS. An example of PoA is Peercoin (PPC) that uses an initial PoW consensus mechanism combined with a PoS consensus. By integrating the PoW and PoS consensus mechanisms, a lower amount of energy is required, and the probability of 51% attack is also reduced. A 51% attack is basically a common attack on a blockchain network, where a single node or organization takes the control of the majority of the hash rate and causes disruption in the blockchain

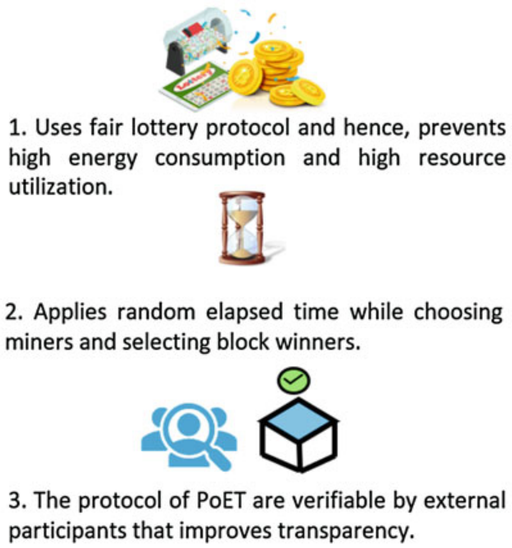
1. Uses fair lottery protocol and hence, prevents high energy consumption and high resource utilization.
2. Applies random elapsed time while choosing miners and selecting block winners.
3. The protocol of PoET are verifiable by external participants that improves transparency.

Fig. 2.7 Proof-of-elapsed-time (PoET) consensus mechanism

network. In such a situation, the attacker who grabs control over the majority hash rate can intentionally modify or exclude any transactions from the blockchain. The 51% attack problem is mostly seen in PoW consensus model, and PoA extensively reduces the chances of such an attack. In Fig. 2.8, we presented an illustration of proof-of-activity (PoA) consensus mechanism.

2.2.6 *Proof of Importance*

Proof-of-importance (PoI) consensus model is used to determine eligible users to perform necessary calculations while adding a new block to a blockchain and collecting associated payment. A PoI consensus model prioritizes blockchain miners by considering the value of coins it contains and the number of transactions they perform in their corresponding cryptocurrency. The more the transactions are made from the nodes' wallet, the higher the chances of being given to create the next block. In Fig. 2.9, we presented an illustration of proof-of-importance (PoI) consensus mechanism.

2.2.7 *Proof of Capacity*

The idea of proof of capacity (PoC) was first introduced in [18]. In this approach, the miners use free spaces of their hard disk for mining free coins. The algorithm comprises plotting hard drive of the nodes, i.e., performing computation and storing solutions on the hard disks before the block mining begins. The node that stores the fastest (closest) solution of the block puzzle wins the block. Using this protocol,

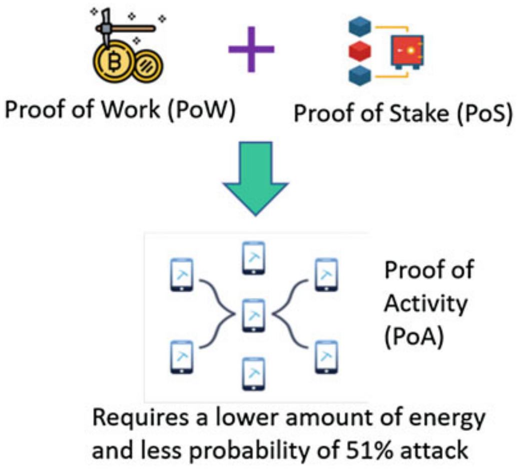
Proof of Work (PoW)
+
Proof of Stake (PoS)
Proof of Activity (PoA)
Requires a lower amount of energy and less probability of 51% attack

Fig. 2.8 Proof-of-activity (PoA) consensus mechanism

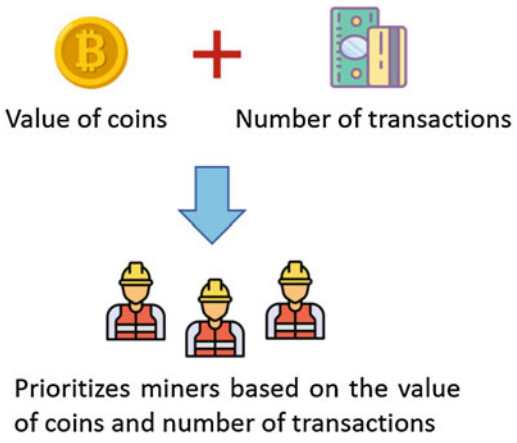
Value of coins
Number of transactions
Prioritizes miners based on the value of coins and number of transactions

Fig. 2.9 Proof-of-importance (PoI) consensus mechanism

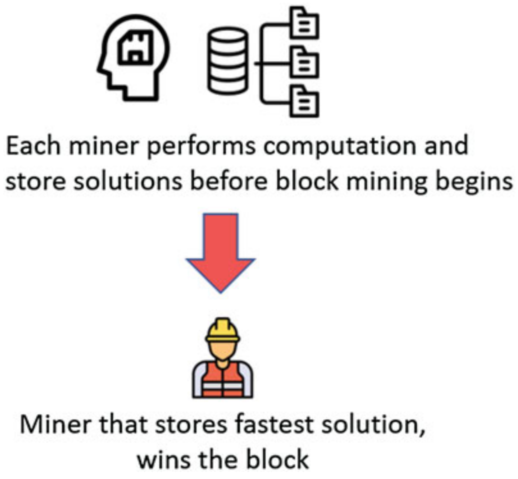
Each miner performs computation and store solutions before block mining begins
Miner that stores fastest solution, wins the block

Fig. 2.10 Proof-of-capacity (PoC) consensus mechanism

the capacity of storage space of user's hard drive can be utilized. In this consensus model, the more space in hard drive a node has, the better chance to mine the next block and get reward. In Fig. 2.10, we presented an illustration of proof-of-capacity (PoC) consensus mechanism.

2.2.8 Proof of Burn

In proof-of-burn (PoB) consensus model, the blockchain nodes "burn" their coins, i.e., the nodes send their coins to an address from where the coins are irretrievable. By passing the coins toward a never–never land, each node gets a lifetime privilege to perform system mining through a random selection procedure [19]. Blockchain miners have a better chance and prioritize mining the next block as per the amount

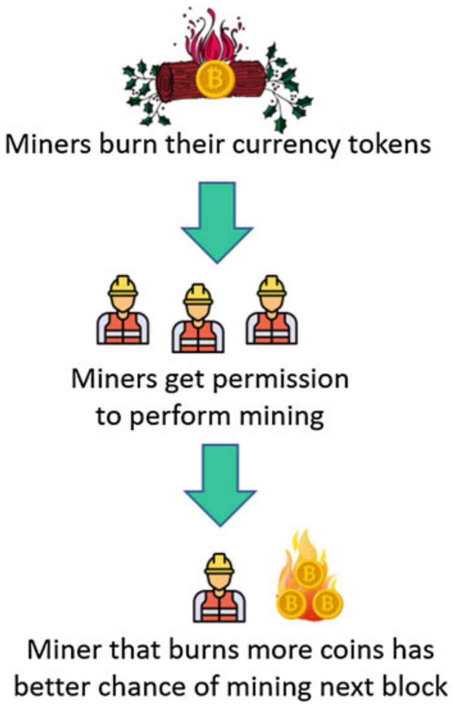
Miners burn their currency tokens
Miners get permission to perform mining
Miner that burns more coins has better chance of mining next block

Fig. 2.11 Proof-of-burn (PoB) consensus mechanism

of coins they burn. In Fig. 2.11, we presented an illustration of proof-of-burn (PoB) consensus mechanism.

2.3 Comparison of Typical Consensus Algorithms

In this section, we compare the consensus algorithms that we discussed in Sect. 2.2 (Table 2.1). We compare the consensus algorithm based on accessibility, scalability, decentralization, throughput, and latency [15, 20, 21].

2.4 Discussion

This chapter covers the life cycle and a detailed working procedure of smart contracts in blockchain technology. We also explain various consensus algorithms, including their working process, and discuss what criteria are required while applying those consensus mechanisms in blockchain technology. To the end, we show a comparison table that highlights the core difference among those consensus algorithms.

Table 2.1 Comparison of consensus algorithms

Consensus method	Accessibility	Scalability	Decentralization	Throughput	Latency
PoW	Public, permissionless	High	High	Low	High
PoS	Public, permissionless, permissioned	High	High	Low	High
PBFT	Private, permissioned	Low	Medium	High	Low
PoET	Public, permissionless, permissioned	High	Medium	High	Low
PoA	Public, permissionless	High	High	Low	Medium
PoI	Public, permissionless	High	High	High	Medium
PoC	Public, permissionless	High	High	Low	High
PoB	Public, permissionless	High	High	Low	High

References

1. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. Manubot (2019)
2. Szabo, N.: The idea of smart contracts. Nick Szabo's Papers and Concise Tutorials 6 (1997)
3. Wikipedia Contributors: Smart contract, Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Smart_contract&oldid=983355857 (accessed October 29, 2020)
4. Dai, H.-N., Zheng, Z., Zhang, Y.: Blockchain for Internet of Things: A survey. IEEE Int. Things J. **6**(5), 8076–8094 (2019)
5. Zheng, Z., Xie, S., Dai, H.-N., Chen, W., Chen, X., Weng, J., Imran, M.: An overview on smart contracts: Challenges, advances and platforms. Future Gener. Comput. Syst. **105**, 475–491 (2020). https://doi.org/10.1016/j.future.2019.12.019
6. Idelberger, F., Governatori, G., Riveret, R., Sartor, G.: Evaluation of logic-based smart contracts for blockchain systems. In: International Symposium on Rules and Rule Markup Languages for the Semantic Web, RuleML, pp. 167–183. Springer (2016)
7. Sillaber, C., Waltl, B.: Life cycle of smart contracts in blockchain ecosystems. Datenschutz und Datensicherheit-DuD **41**(8), 497–500 (2017)
8. Salimitari, M., Chatterjee, M.: A survey on consensus protocols in blockchain for iot networks. Preprint (2018). arXiv:1809.05613
9. Zhou, Q., Huang, H., Zheng, Z., Bian, J.: Solutions to scalability of blockchain: A survey. IEEE Access **8**, 16440–16455 (2020)
10. Xu, C., Wang, K., Guo, M.: Intelligent resource management in blockchain-based cloud datacenters. IEEE Cloud Comput. **4**(6), 50–59 (2017)
11. Wang, W., Hoang, D.T., Xiong, Z., Niyato, D., Wang, P., Hu, P., Wen, Y.: A survey on consensus mechanisms and mining management in blockchain networks. Preprint, 1–33 (2018). arXiv:1805.02707
12. Zheng, Z., Xie, S., Dai, H., Chen, X., Wang, H.: An overview of blockchain technology: Architecture, consensus, and future trends. In: 2017 IEEE International Congress on Big Data (BigData Congress), pp. 557–564. IEEE (2017)

13. Bashir, I.: Mastering Blockchain. Packt Publishing Ltd. (2017)
14. Bach, L.M., Mihaljevic, B., Zagar, M.: Comparative analysis of blockchain consensus algorithms. In: 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1545–1550. IEEE (2018)
15. Salimitari, M., Chatterjee, M.: A survey on consensus protocols in blockchain for iot networks. Preprint (2018). arXiv:1809.05613
16. Bamakan, S.M.H., Motavali, A., Bondarti, A.B.: A survey of blockchain consensus algorithms performance evaluation criteria. Expert Syst. Appl. 113385 (2020)
17. Wang, W., Hoang, D.T., Hu, P., Xiong, Z., Niyato, D., Wang, P., Wen, Y., Kim, D.I.: A survey on consensus mechanisms and mining strategy management in blockchain networks. IEEE Access **7**, 22328–22370 (2019)
18. Dziembowski, S., Faust, S., Kolmogorov, V., Pietrzak, K.: Proofs of space. Paper presented at the Annual Cryptology Conference (2015)
19. Debus, J.: Consensus methods in blockchain systems. Frankfurt School of Finance & Management, Blockchain Center, Tech. Rep (2017)
20. Chalaemwongwan, N., Kurutach, W.: State of the art and challenges facing consensus protocols on blockchain. In: 2018 International Conference on Information Networking (ICOIN), pp. 957–962. IEEE (2018)
21. Zamani, M., Movahedi, M., Raykova, M.: RapidChain: Scaling blockchain via full sharding. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, pp. 931–948. ACM (2018)