

Claims:

1. A data structure embodied in one or more blockchain transactions held in transitory or non-transitory computer-readable media, the data structure having:
 - 5 a plurality of nodes, each node embodied as a hash value contained in a blockchain transaction of the one or more blockchain transactions; and
 - a plurality of directional edges;
 - wherein the plurality of nodes comprises leaf nodes and non-leaf nodes, every non-leaf node having at least one child node directly connected thereto by a directional edge,
 - 10 and every child node being a non-leaf node or a leaf node without any child node connected thereto, the non-leaf nodes including a common root node to which all other nodes are connected directly or indirectly via one or more of the non-leaf nodes;
 - wherein the hash value of each non-leaf node is a hash of a concatenation of the hash values of all of its child node(s), and wherein the hash value of each leaf node is a hash
 - 15 of an external data block; and
 - wherein at least one of the non-leaf nodes has at least one child leaf node and at least one child non-leaf node, the hash value of the at least one non-leaf node being a hash of a concatenation of the respective hash values of the child leaf node and the child non-leaf node.
- 20 2. The data structure of claim 1, wherein a first of the non-leaf nodes has a different number of child nodes connected thereto than a second of the non-leaf nodes.
3. The data structure of claim 1 or 2, wherein a first of the leaf nodes has a different
- 25 level than a second of the leaf nodes, the level of each node being the number of directional edges via which the node is directly or indirectly connected to the common root node.
4. A data structure embodied in one or more blockchain transactions held in transitory or non-transitory computer-readable media, the data structure having:
 - 30 a plurality of nodes, each node embodied as a hash value contained in a blockchain transaction of the one or more blockchain transactions; and
 - a plurality of directional edges;

wherein the plurality of nodes comprises leaf nodes and non-leaf nodes, every non-leaf node having at least one child node directly connected thereto by a directional edge, and every child node being a non-leaf node or a leaf node without any child node connected thereto, the non-leaf nodes including a common root node to which all other nodes are
5 connected directly or indirectly via one or more of the non-leaf nodes;

wherein the hash value of each non-leaf node is a hash of a concatenation of the hash values of all of its child node(s), and wherein the hash value of each leaf node is a hash of an external data block; and

wherein a first of the non-leaf nodes has a different number of child nodes than a
10 second of the non-leaf nodes.

5. A data structure embodied in one or more blockchain transactions held in transitory or non-transitory computer-readable media, the data structure having:

a plurality of nodes, each node embodied as a hash value contained in a blockchain
15 transaction of the one or more blockchain transactions; and

a plurality of directional edges;

wherein the plurality of nodes comprises leaf nodes and non-leaf nodes, every non-leaf node having at least one child node directly connected thereto by a directional edge, and every child node being a non-leaf node or a leaf node without any child node connected
20 thereto, the non-leaf nodes including a common root node to which all other nodes are connected directly or indirectly via one or more of the non-leaf nodes;

wherein the hash value of each non-leaf node is a hash of a concatenation of the hash values of all of its child node(s), and wherein the hash value of each leaf node is a hash of an external data block; and

25 wherein a first of the leaf nodes has a different level than a second of the leaf nodes, the level of each node being the number of directional edges via which the node is directly or indirectly connected to the common root node.

6. The data structure of any preceding claim, wherein:

30 (i) each node directly or indirectly connected to the common root node is associated with a sibling index denoting a position of the node relative to any sibling nodes thereof, sibling nodes being child nodes of a common parent node; and

(ii) each node indirectly connected to the common root node is associated with one or more intermediary indices identifying the one or more non-leaf nodes via which the node is indirectly connected to the common root node.

5 7. The data structure of claim 6, wherein the index or indices associate with each node are directly encoded in the one or more blockchain transactions.

8. The data structure of claim 6, wherein the index or indices associate with each node are not directly encoded in the one or more blockchain transactions but are stored in an off-
10 chain data store.

9. The data structure of any preceding claim, wherein the hash value of each leaf node is a double hash or other multi-hash of the external data block.

15 10. A computer-implemented method of creating or updating the data structure of any preceding claim, the method comprising:

receiving an external data block to be represented in the data structure;

applying at least one hash function to the external data block to compute a hash
value therefrom; and

20 generating or modifying a blockchain transaction of the one or more blockchain transactions, the generated or modified blockchain transaction containing the hash value, and thereby creating a leaf node, within the data structure, representing the received external data block.

25 11. The method of claim 10, the steps of which are performed for each leaf node of the data structure so as to create the data structure.

12. The method of claim 10 or 11, comprising the step of transmitting the blockchain transaction to a node of a blockchain network for causing the node to process the
30 blockchain transaction for recording in a blockchain.

13. The method of claim 10 or 11, comprising the step of sending the blockchain transaction to an off-chain system for processing.

14. A computer-implemented method of verifying a received data block using the data structure of any of claims 1 to 7, the method comprising:

receiving the data block to be verified, the received data block corresponding to one of the leaf nodes;

applying at least one hash function to the received data block, thereby computing a reconstructed leaf node hash;

determining, from the data structure, an authentication path for the external data block, the authentication path being a set of one or more of the nodes required to reconstruct the hash value of the common root node;

computing a reconstructed root node hash using the reconstructed leaf node hash and the hash value(s) of the one or more nodes of the authentication path, by applying

successive hashing and concatenating operations in accordance with the directional edges between those nodes; and

comparing the reconstructed root node hash with the hash value of the common root node, and thereby verifying the received data block.

15. The method of any of claims 10 to 14, comprising calculating:

(i) for each node directly or indirectly connected to the common root node, a sibling index denoting a position of the node relative to any sibling nodes thereof, sibling nodes being child nodes of a common parent node, and

(ii) for each node indirectly connected to the common root node, one or more intermediary indices identifying the one or more non-leaf nodes via which the node is indirectly connected to the common root node.

16. The method of claim 15, wherein the index or indices calculated for each node are directly encoded in the one or more blockchain transactions.

17. The method of claim 15, wherein the calculated indices are not directly encoded in the one or more blockchain transactions but are stored in an off-chain datastore.

18. The method of any of claims 15 to 17 when dependent on claim 10, wherein, in order to create the data structure, the hash value $N_{0,i_0,\dots,i_{m-2},j}$ of each non-leaf node is computed as:

$$5 \quad N_{0,i_0,\dots,i_{m-2},j} = H \left(\sum_{\alpha=0}^{n-1} N_{0,i_0,\dots,i_{m-2},j,\alpha} \right)$$

wherein $i_0, \dots, i_{m-2}$ represents any intermediary index or indices of the non-leaf node, $N_{0,i_0,\dots,i_{m-2},j,\alpha}$ is the hash value of a child node of the non-leaf node with $i_0, \dots, i_{m-2}, j$ representing the one or more intermediary indices of the child node, j being the sibling index of the non-leaf node and also the final intermediary index of the child node, and α being the child node's sibling index;

wherein $\sum_{\alpha=0}^{n-1} N_{0,i_0,\dots,i_{m-2},j,\alpha}$ denotes concatenation of the hash values of all child nodes of the non-leaf node; and

wherein H is a hash function.

15 19. The method of any of claims 15 to 18 when dependent on claim 14, wherein the index or indices calculated for the node(s) of the authentication path and the node corresponding to the received data block are used in computing the reconstructed root node hash, by computing the hash value $N_{0,i_0,\dots,i_{m-2},j}$ of each node of the authentication path is as:

$$20 \quad N_{0,i_0,\dots,i_{m-2},j} = H \left(\sum_{\alpha=0}^{n-1} N_{0,i_0,\dots,i_{m-2},j,\alpha} \right),$$

wherein $i_0, \dots, i_{m-2}$ represents any intermediary index or indices of the non-leaf node, $N_{0,i_0,\dots,i_{m-2},j,\alpha}$ i:

the hash value of a child node of the non-leaf node if the child node forms part of the authentication path, and

25 the reconstructed leaf node hash if the child node is the leaf node corresponding to the received data block to be verified,

with $i_0, \dots, i_{m-2}, j$ representing the one or more intermediary indices of the child node, j being the sibling index of the non-leaf node and also the final intermediary index of the child node, and α being the child node's sibling index;

wherein $\sum_{\alpha=0}^{n-1} N_{0,i_0,\dots,i_{m-2},j,\alpha}$ denotes concatenation of the hash values of all child nodes of the non-leaf node; and

wherein H is a hash function.

- 5 20. A computer system comprising one or more computer processors and computer-readable media coupled to the one or more computer processors for embodying the data structure of any of claims 1 to 13, wherein the one or more computer processors are configured to implement the method of any of claims 14 to 19.
- 10 21. Computer-readable program instructions embodied on transitory or non-transitory media and configured, when executed on one or more computer processors, to implement the method of any of claims 14 to 19.