

CLAIMS

1. A computer-implemented method of creating different versions of a tree structure overlaid on a blockchain, the tree structure comprising a set of nodes and edges between nodes, wherein each node is a different transaction recorded on the blockchain, wherein each edge connects from a respective child node to a respective parent node, wherein one of the parent nodes is a root node of the tree structure, wherein each node is associated with a respective key, wherein each child node comprises i) a respective transaction identifier, and ii) a signature corresponding to the respective key associated with the respective parent node, and wherein the method is performed by a tree creator and comprises:

creating one or more target child nodes of a target parent node, each target child node comprising a respective data payload; and

forming a respective edge between each target child node and the target parent node by associating each of the target child nodes with a respective link identifier, wherein the respective link identifier is based on at least one off-chain parameter.

2. The method of claim 1, wherein at least two of the target child nodes are associated with different respective link identifiers.

3. The method of claim 1 or claim 2, wherein each of the target child nodes are associated with a different respective link identifier.

4. The method of any preceding claim, comprising receiving one, some or each of the respective link identifiers from one or more entities other than the tree creator.

5. The method of any preceding claim, wherein one, some or each of the respective link identifiers are generated by supplying a respective set of off-chain parameters to a respective link identifier function that is configured to generate a link identifier based on a set of parameters.

6. The method of claim 5, wherein one, some or each of the respective link identifiers are generated by supplying a respective set of off-chain parameters to a same link identifier function.
7. The method of claim 6, wherein at least two of the respective link identifiers are generated by supplying a respective set of off-chain parameters to a different link identifier function.
8. The method of claim 5 or any claim dependent thereon, wherein the respective set of off-chain parameters supplied to the respective link identifier function comprises one, some or each of the following:
 - one or more time-related parameters,
 - one or more user-specific parameters,
 - one or more parameters specific to a group of users,
 - one or more weighting parameters,
 - one or more dataset-specific parameters, and
 - one or more application-specific parameters.
9. The method of claim 8, wherein in addition to the respective set of off-chain parameters, a transaction identifier of the target parent node is also supplied to the respective link identifier function to generate the respective link identifier.
10. The method of claim 5 or any claim dependent thereon, wherein one or more of the respective set of parameters supplied to the link identifier function are encrypted.
11. The method of claim 5 or any claim dependent thereon, wherein one, some or each of the respective link identifiers are generated based on a different value of at least one same parameter.
12. The method of claim 5 or any claim dependent thereon, wherein the link identifier function comprises a hash function.

13. The method of any preceding claim, wherein one, some or each of the target child nodes are associated with a respective user.
14. The method of claim 13 when dependent on claim 5 comprising receiving one or more respective link identifiers from one or more respective users.
15. The method of claim 13 when dependent on claim 5, comprising receiving one or more respective sets of parameters from one or more respective users.
16. The method of claim 5 or any claim dependent thereon, comprising embedding proof-of-work into the respective edge between at least one target child node and the parent node by generating a respective link identifier satisfying a predetermined difficulty, wherein the respective link identifier is generated by supplying the respective set of parameters and a nonce value to the respective link identifier function.
17. A computer-implemented method of accessing a tree structure overlaid on a blockchain, the tree structure comprising a set of nodes and edges between nodes, wherein each node is a different transaction recorded on the blockchain, wherein each edge connects from a respective child node to a respective parent node, wherein one of the parent nodes is a root node of the tree structure, wherein each node is associated with a respective key, wherein each child node comprises i) a respective transaction identifier, and ii) a signature corresponding to the respective key associated with the respective parent node, wherein a target parent node is connected to a plurality of target child nodes, each target child node comprising a respective data payload, wherein each of the target child nodes is associated with a respective link identifier, wherein the respective link identifier is based on at least one off-chain parameter, and wherein the method is performed by a tree accessor and comprises:
 - obtaining the target parent node;
 - obtaining one or more link identifiers;
 - identifying one or more of the target child nodes that are associated with a respective one of the obtained one or more link identifiers; and

creating a version of the tree structure comprising one or more of the identified target child nodes, but not those target child nodes that are not identified as being associated with a respective one of the obtained one or more link identifiers.

18. The method of claim 17, comprising:

for one or more of the target child nodes forming the created version of the tree structure, performing at least one of: accessing, storing and/or using the respective data payload comprised by that target child node.

19. A computer system comprising:

processing apparatus comprising one or more processing units, and

memory comprising one or more memory units;

wherein the memory stores code arranged to run on the processing apparatus, the code being configured so as when run on the processing apparatus to perform operations in accordance with any preceding claim.

20. A computer program embodied on computer-readable storage, the computer program comprising code configured so as when run on one or more processing units to perform operations in accordance with any of claims 1 to 18.