

CLAIMS

~~1.~~ 1. A system (300) for processing world state information associated with a blockchain, the system comprising:

a blockchain database (322) configured to store a blockchain (216) that is managed by a blockchain network, (102, 212, 314, 316), in which blockchain data are stored on the blockchain through consensus of blockchain nodes (214) of the blockchain network, and the blockchain includes one or more blocks that store transaction data associated with user accounts associated with the blockchain;

a world state database (324) configured to store a world state data structure comprising a plurality of digital trees, in which the world state (310a, 312a, 310b, 312b, 310c, 312c) stores information about states of the user accounts;

~~_____~~ a smart contract database storing a plurality of smart contracts associated with the blockchain, in which a first smart contract among the plurality of smart contracts comprises first code that identifies first state data and a first digital tree among the plurality of digital trees for storing the first state data; ~~and, and in which the first smart contract comprises second code that identifies second state data and a second digital tree among the plurality of digital trees for storing the second state data; and~~

a virtual machine associated with the blockchain corresponding to the first smart contract, in which the virtual machine is configured to execute the first smart contract to cause the first state data to be stored in the first digital tree.

~~2. The system of claim 1 in which and the second state data to be stored in the second digital tree, wherein the system or~~ the virtual machine includes a smart contract compiler configured to compile the first smart contract into first bytecode that can be executed by the virtual machine, ~~in which and wherein~~ the compiler is configured to recognize and compile the first code ~~such that and the first bytecode when executed by the virtual machine causes the first state data to be stored in the first digital tree.~~

~~3. The system of claim 1, comprising a smart contract compiler configured to compile the first smart contract into first bytecode that can be executed by the virtual machine, in which the compiler is configured to recognize and compile the firstsecond code such that the first bytecode when executed by the virtual machine causes the first state data to be stored in the first digital tree; and the second state data to be stored in the second digital tree.~~

4. ~~2.~~ The system (300) of ~~any one of claims~~claim 1 to 3 in which the state data identified by the first code comprise at least one of (i) data about a state of a user account associated with the first smart contract, or (ii) data about a state of the first smart contract.

~~5. The system of claim 2 or 3 in which the first smart contract comprises second code that identifies second state data and a second digital tree among the plurality of digital trees for storing the second state data;~~

~~wherein the compiler is configured to recognize and compile the first code and the second code such that the first bytecode when executed by the virtual machine causes the first state data to be stored in the first digital tree and the second state data to be stored in the second digital tree.~~

~~The system of claim 5~~

6. ~~3.~~ The system (300) of ~~any one of claims 1 or 2~~ in which the first and second digital trees in combination maintain a world state that is associated with user accounts associated with the first smart contract; and/or

~~The system of claim 5 or 6~~ in which the first code specifies that state data associated with accounts of users in a first

~~7.~~ geographical region are to be stored in the first digital tree, and the second code specifies that state data associated with accounts of users in a second geographical region are to be stored in the second digital tree.

8. ~~4.~~ The system (300) of ~~claim 5~~any one of claims 1 or 2 in which the first smart contract comprises third code that identifies third state data and a third digital tree among the plurality of digital trees for storing the third state data;

wherein the first, second, and third digital trees in combination maintain the world state that is associated with user accounts associated with the first smart contract;

 wherein the compiler is configured to recognize and compile the first code, the second code, and the third code such that the first bytecode when executed by the virtual machine causes the first state data to be stored in the first digital tree, the second state data to be stored in the second digital tree, and the third state data to be stored in the third digital tree.

~~9.~~ 5. The system (300) of claim ~~84~~ in which the first, second, and third digital trees in combination maintain a world state that is associated with user accounts associated with the first smart contract; and/or

~~10.~~ ~~The system of claim 8 or 9~~ in which the first code specifies that state data associated with accounts of users in a first geographical region are to be stored in the first digital tree, the second code specifies that state data associated with accounts of users in a second geographical region are to be stored in the second digital tree, and the third code specifies that state data associated with accounts of users in a third geographical region are to be stored in the third digital tree.

~~11.~~ 6. The system (300) of any one of claims ~~21~~ to ~~105~~ in which a second smart contract among the plurality of smart contracts comprises:

fourth code that identifies fourth state data and a fourth digital tree among the plurality of digital trees for storing the fourth state data;

 wherein the first and second smart contracts are associated with a same blockchain, and the first and fourth digital trees in combination maintain a world state that stores state data of user accounts associated with the first and second smart contracts;

wherein the compiler is configured to compile the second smart contract into second bytecode; and

 wherein the compiler is configured to recognize and compile the fourth code such that the second bytecode when executed by the virtual machine causes the fourth state data to be stored in the fourth digital tree.

~~12.~~ 7. The system (300) of any one of claims ~~21~~ to ~~116~~ in which the first code comprises a first address of the first digital tree, and the compiler is configured to compile the first smart contract into the first bytecode such that the first bytecode when executed by the virtual machine causes the first state data to be stored in the first digital tree having the first address.

~~13.~~ ~~8.~~ The system (300) of claim ~~127~~ in which the first address is represented by a constant value;

~~14. The system of claim 12; or~~ in which the first address is determined by at least one of a Boolean function or a mathematical function;

~~15. The system of claim 12; or~~ in which the first address is determined by a rule set comprising one or more rules;

~~16. The system of claim 12; or~~ in which the first address is determined at a first time period when the first smart contract is compiled, or at a second time period when the first bytecode is executed.

17. 9. The system (300) of any one of claims ~~51~~ to ~~168~~ in which the second code comprises a second address of the second digital tree, and the compiler is configured to compile the first smart contract into the first bytecode such that the first bytecode when executed by the virtual machine causes the second state data to be stored in the second digital tree having the second address; optionally in which at least one of:

~~18. The system of claim 17 in which at least one of:~~

~~(i)~~ (i) the second address is represented by a constant value;

~~(ii)~~ (ii) the second address is determined by a Boolean function;

~~(iii)~~ (iii) the second address is determined by a mathematical function;

~~(iv)~~ (iv) the second address is determined by a rule set comprising one or more rules;

~~(v)~~ (v) the second address is determined at a first time period when the second smart contract is compiled; or

~~(vi)~~ (vi) the second address is determined at a second time period when the second bytecode is executed.

19. 10. The system (300) of any one of claims ~~84~~ to ~~189~~ in which the third code comprises a third address of the third digital tree, and the compiler is configured to compile the first smart contract into the first bytecode such that the first bytecode when executed by the virtual machine causes the third state data to be stored in the third digital tree having the third address.

~~20. 11. The system (300) of any one of claims ~~116~~ to ~~1910~~~~ in which the fourth code comprises a fourth address of the fourth digital tree, and the compiler is configured to compile

the second smart contract into the second bytecode such that the second bytecode when executed by the virtual machine causes the fourth state data to be stored in the fourth digital tree having the fourth address.

~~21.~~ 12. The system (300) of any one of claims ~~51~~ to ~~1811~~, comprising one or more computers in one or more locations and configured to:

 determine a first root hash value of the first digital tree;
 determine a second root hash value of the second digital tree; and
 determine a world state root hash value based on the first root hash value and the second root hash value;

~~22.~~ The system of claim 21; optionally in which the one or more computers are configured to:

 determine a third root hash value of the third digital tree; and
 determine the world state root hash value based on the first root hash value, the second root hash value, and the third root hash value;

~~23.~~ The system of claim 21 or 22; and/or in which the world state root hash value represents a digital fingerprint of the state or states of the user accounts, smart contract, or smart contracts associated with the world state.

~~24.~~ 13. The system (300) of any one of claims ~~51~~ to ~~2312~~ in which the first smart contract specifies that the first digital tree be stored at a first storage device, and the second digital tree be stored at a second storage device.

~~25.~~ 14. The system (300) of any one of claims ~~51~~ to ~~2413~~ in which the first smart contract specifies that both the first digital tree and the second digital tree be stored at a first storage device.

~~26.—~~15. The system (300) of ~~claim 25, comprising any one of claims 1 to 14, in which~~ the system comprises one or more computers in one or more locations and configured to generate an index of the data stored in the first storage device, and store the index in the first and second digital trees;

~~27. — The system of any one of claims 1 to 26; and/or~~ in which each digital tree comprises at least one of a Merkle tree, a Patricia tree, a Merkle Patricia tree, a modified Merkle Patricia tree, or a Bucket tree.

~~28. — A method for processing world state information associated with a blockchain, the method comprising:~~

~~providing a plurality of smart contracts associated with a plurality of user accounts, in which each smart contract is associated with a corresponding blockchain managed by a corresponding blockchain network, and blockchain data are stored on the blockchain through consensus of blockchain nodes of the blockchain network;~~

~~providing a plurality of digital trees to maintain one or more world states that are associated with the smart contracts and the user accounts, in which each world state stores state data associated with at least one of a user account or a smart contract;~~

~~wherein a first smart contract among the plurality of smart contracts comprises:~~

~~first code that identifies first state data and a first digital tree among the plurality of digital trees for storing the first state data; and~~

~~providing a virtual machine associated with the blockchain corresponding to the first smart contract, in which the virtual machine is configured to execute the first smart contract to cause the first state data to be stored in the first digital tree.~~

~~29. — The method of claim 28 in which the virtual machine includes a smart contract compiler configured to compile the first smart contract into first bytecode that can be executed by the virtual machine, in which the compiler is configured to recognize and compile the first code such that the first bytecode when executed by the virtual machine causes the first state data to be stored in the first digital tree.~~

~~30. — The method of claim 28, comprising providing a smart contract compiler configured to compile the first smart contract into first bytecode that can be executed by the virtual machine, in which the compiler is configured to recognize and compile the first code such that the first bytecode when executed by the virtual machine causes the first state data to be stored in the first digital tree.~~

~~31.—The method of any one of claims 28 to 30 in which the state data identified by the first code comprise at least one of (i) data about a state of a user account associated with the first smart contract, or (ii) data about a state of the first smart contract.~~

~~32.—The method of any one of claims 29 to 31 in which the first smart contract comprises second code that identifies second state data and a second digital tree among the plurality of digital trees for storing the second state data;~~

~~wherein the compiler is configured to recognize and compile the first code and the second code such that the first bytecode when executed by the virtual machine causes the first state data to be stored in the first digital tree and the second state data to be stored in the second digital tree.~~

~~33.—The method of claim 32 in which the first and second digital trees in combination maintain a world state that is associated with user accounts associated with the first smart contract.~~

~~34.—The method of claim 32 or 33 in which the first code specifies that state data associated with accounts of users in a first geographical region are to be stored in the first digital tree, and the second code specifies that state data associated with accounts of users in a second geographical region are to be stored in the second digital tree.~~

~~35.—The method of claim 32 in which the first smart contract comprises third code that identifies third state data and a third digital tree among the plurality of digital trees for storing the third state data;~~

~~wherein the first, second, and third digital trees in combination maintain the world state that is associated with user accounts associated with the first smart contract;~~

~~wherein the compiler is configured to recognize and compile the first code, the second code, and the third code such that the first bytecode when executed by the virtual machine causes the first state data to be stored in the first digital tree, the second state data to be stored in the second digital tree, and the third state data to be stored in the third digital tree.~~

~~36.—The method of claim 35 in which the first, second, and third digital trees in combination maintain a world state that is associated with user accounts associated with the first smart contract.~~

~~37.—The method of claim 35 or 36 in which the first code specifies that state data associated with accounts of users in a first geographical region are to be stored in the first digital tree, the second code specifies that state data associated with accounts of users in a second geographical region are to be stored in the second digital tree, and the third code specifies that state data associated with accounts of users in a third geographical region are to be stored in the third digital tree.~~

~~38.—The method of any one of claims 29 to 37 in which a second smart contract among the plurality of smart contracts comprises:~~

~~fourth code that identifies fourth state data and a fourth digital tree among the plurality of digital trees for storing the fourth state data;~~

~~wherein the first and second smart contracts are associated with a same blockchain, and the first and fourth digital trees in combination maintain a world state that stores state data of user accounts associated with the first and second smart contracts;~~

~~wherein the compiler is configured to compile the second smart contract into second bytecode; and~~

~~wherein the compiler is configured to recognize and compile the fourth code such that the second bytecode when executed by the virtual machine causes the fourth state data to be stored in the fourth digital tree.~~

~~39.—The method of any one of claims 29 to 38 in which the first code comprises a first address of the first digital tree, and the compiler is configured to compile the first smart contract into the first bytecode such that the first bytecode when executed by the virtual machine causes the first state data to be stored in the first digital tree having the first address.~~

~~40.—The method of claim 39 in which the first address is represented by a constant value.~~

~~41.—The method of claim 39 in which the first address is determined by at least one of a Boolean function or a mathematical function.~~

~~42.—The method of claim 39 in which the first address is determined by a rule set comprising one or more rules.~~

~~43.—The method of claim 39 in which the first address is determined at a first time period when the first smart contract is compiled, or at a second time period when the first bytecode is executed.~~

~~44.—The method of any one of claims 32 to 43 in which the second code comprises a second address of the second digital tree, and the compiler is configured to compile the first smart contract into the first bytecode such that the first bytecode when executed by the virtual machine causes the second state data to be stored in the second digital tree having the second address.~~

~~45.—The method of claim 44 in which at least one of:~~

- ~~(vii)—the second address is represented by a constant value;~~
- ~~(viii)—the second address is determined by a Boolean function;~~
- ~~(ix)—the second address is determined by a mathematical function;~~
- ~~(x)—the second address is determined by a rule set comprising one or more rules;~~
- ~~(xi)—the second address is determined at a first time period when the second smart contract is compiled; or~~
- ~~(xii)—the second address is determined at a second time period when the second bytecode is executed.~~

~~46.—The method of any one of claims 35 to 45 in which the third code comprises a third address of the third digital tree, and the compiler is configured to compile the first smart contract into the first bytecode such that the first bytecode when executed by the virtual machine causes the third state data to be stored in the third digital tree having the third address.~~

~~47.—The method of any one of claims 38 to 46 in which the fourth code comprises a fourth address of the fourth digital tree, and the compiler is configured to compile the second smart contract into the second bytecode such that the second bytecode when executed by the virtual machine causes the fourth state data to be stored in the fourth digital tree having the fourth address.~~

~~48.—The method of any one of claims 29 to 47, comprising executing the compiler to generate at least one of the first bytecode or the second bytecode.~~

~~49.—The method of any one of claims 32 to 45, comprising executing the virtual machine and performing at least one of storing the first state data in the first digital tree, storing the second state data in the second digital tree, storing the third state data in the third digital tree, or storing the fourth state data in the fourth digital tree.~~

~~50.—The method of claim 49, comprising:
determining a first root hash value of the first digital tree;
determining a second root hash value of the second digital tree; and
determining a world state root hash value based on the first root hash value and the second root hash value.~~

~~51.—The method of claim 50, comprising:
determining a third root hash value of the third digital tree; and
determining the world state root hash value based on the first root hash value, the second root hash value, and the third root hash value.~~

~~52.—The method of claim 50 or 51 in which the world state root hash value represents a digital fingerprint of the state or states of the user accounts, smart contract, or smart contracts associated with the world state.~~

~~53.—The method of any one of claims 32 to 52 in which the first smart contract specifies that the first digital tree be stored at a first storage device, and the second digital tree be stored at a second storage device.~~

~~54.—The method of any one of claims 32 to 52 in which the first smart contract specifies that both the first digital tree and the second digital tree be stored at a first storage device.~~

~~55.—The method of claim 54, comprising generating an index of the data stored in the first storage device, and storing the index in the first and second digital trees.~~

~~56.—The method of any one of claims 28 to 55 in which each digital tree comprises at least one of a Merkle tree, a Patricia tree, a Merkle Patricia tree, a modified Merkle Patricia tree, or a Bucket tree.~~

~~57.—A method for processing world state information associated with a blockchain, the method comprising:~~

~~providing a first smart contract that is associated with a blockchain managed by a blockchain network in which blockchain data are stored on the blockchain through consensus of blockchain nodes of the blockchain network; and~~

~~providing a first Merkle tree and a second Merkle tree to maintain a first world state that is associated with a first set of user accounts, in which the first world state stores data about current states of the first set of user accounts and current states of the first smart contract;~~

~~wherein the first smart contract comprises:~~

~~first code that specifies first data to be stored in the first Merkle tree, and~~

~~second code that specifies second data to be stored in the second Merkle tree.~~

~~58.—The method of claim 57 in which the first code specifies that data associated with accounts of users in a first geographical region are to be stored in the first Merkle tree, and the second code specifies that data associated with accounts of users in a second geographical region are to be stored in the second Merkle tree.~~

~~59.—The method of claim 57, comprising providing a third Merkle tree, in which the first Merkle tree, the second Merkle tree, and the third Merkle tree in combination maintain a world state that is associated with the user accounts,~~

~~wherein the first smart contract comprises:~~

~~the first code that specifies first data to be stored in the first Merkle tree,~~

~~the second code that specifies second data to be stored in the second Merkle tree, and~~

~~third code that specifies third data to be stored in the third Merkle tree.~~

~~60.—The method of any one of claims 57 to 59 in which the first Merkle tree has a first address, the second Merkle tree has a second address, the first code specifies the first data to be stored in a Merkle tree having the first address, and the second code specifies the second data to be stored in a Merkle tree having the second address.~~

~~61.—The method of any one of claims 57 to 60, comprising providing a virtual machine configured to execute the first smart contract to cause the first data to be stored in the first Merkle tree, and the second data to be stored in the second Merkle tree.~~

~~62. The method of any one of claim 61, comprising providing a compiler configured to compile the first smart contract into bytecodes that can be executed by the virtual machine.~~

~~63. The method of claim 62 in which the compiler is part of the virtual machine.~~

~~64. The method of any one of claims 57 to 61 in which the first smart contract specifies that the first Merkle tree be stored at a first storage device, and the second Merkle tree be stored at a second storage device.~~

~~65. The method of any one of claims 57 to 61 in which in the first smart contract specifies that both the first Merkle tree and the second Merkle tree be stored at a first storage device.~~

~~66. The method of claim 65, comprising generating an index of the data stored in the first storage device, and storing the index in the first and second Merkle trees.~~

~~67. The method of any one of claims 57 to 66, comprising providing a second smart contract that is also associated with the blockchain, in which the second smart contract comprises third code that specifies third data to be stored in the first Merkle tree.~~

~~68. The method of any one of claims 57 to 67 in which the first Merkle tree stores state information of user accounts based in a first geographical region, and the second Merkle tree stores state information of user accounts based in a second geographical region.~~

~~69. The method of any one of claims 57 to 68, comprising providing a second smart contract that is associated with the blockchain;~~

~~providing a third Merkle tree to maintain a second world state that stores data about current states of the second smart contract; and~~

~~determining a world state root hash value based on a first root hash value of the first Merkle tree and a second root hash value of the second Merkle tree.~~

70. ~~A computer implemented method of processing one or more smart contracts associated with a blockchain, the method comprising:~~

~~receiving a smart contract that is associated with a blockchain managed by a blockchain network in which blockchain data are stored on the blockchain through consensus of blockchain nodes of the blockchain network;~~

~~identifying first code in the smart contract that specifies first data and a first Merkle tree;~~

~~identifying second code in the smart contract that specifies second data and a second Merkle tree;~~

~~storing the first data in the first Merkle tree; and~~

~~storing the second data in the second Merkle tree.~~

71. ~~The method of claim 70 in which the first code specifies that data associated with accounts of users in a first geographical region are to be stored in the first Merkle tree, and the second code specifies that data associated with accounts of users in a second geographical region are to be stored in the second Merkle tree.~~

72. ~~The method of claim 70 or 71, comprising executing a virtual machine to execute the smart contract to cause the first data to be stored in the first Merkle tree, and the second data to be stored in the second Merkle tree.~~

73. ~~A method for processing world state information associated with a blockchain, the method comprising:~~

~~providing a smart contract that is associated with a blockchain managed by a blockchain network, in which blockchain data are stored on the blockchain through consensus of blockchain nodes of the blockchain network; and~~

~~providing a plurality of Merkle trees to maintain a world state that is associated with a set of user accounts, in which the world state stores data about current states of the user accounts and current states of the smart contract;~~

~~wherein the smart contract comprises first code that specifies which Merkle tree among the plurality of Merkle trees is used to store a piece of data.~~

74. ~~The method of claim 73 in which the first code comprises a modifier that includes an identifier of a Merkle tree that is configured to store the piece of data.~~

~~75.—The method of claim 74 in which the identifier of the Merkle tree comprises an address of the Merkle tree.~~

~~76.—The method of claim 74 in which the identifier of the Merkle tree comprises a variable.~~

~~77.—The method of any one of claims 73 to 76 in which the identifier is determined based on a mathematical function.~~

~~78.—The method of any one of claims 73 to 76 in which the identifier is determined based on one or more rules.~~

~~79.—The method of any one of claims 74 to 76, comprising providing a compiler configured to recognize the modifier and compiling the smart contract into bytecode that can be executed by a virtual machine associated with the blockchain.~~

~~80.—The method of claim 79 in which the compiler is part of the virtual machine.~~

~~81.—A system comprising:~~

~~one or more processors; and~~

~~—one or more computer-readable memories coupled to the one or more processors and having instructions stored thereon that are executable by the one or more processors to perform the method of any of claims 28 to 80.~~

~~82.—An apparatus comprising a plurality of modules for performing the method of any one of claims 28 to 80.~~

~~83.—A system for processing world state information associated with a blockchain, the system comprising:~~

~~a blockchain database configured to store a blockchain that is managed by a blockchain network, in which blockchain data are stored on the blockchain through consensus of blockchain nodes of the blockchain network, and the blockchain includes one or more blocks that store transaction data associated with user accounts associated with the blockchain;~~

a world state database configured to store a world state data structure comprising a first Merkle tree and a second Merkle tree, in which the world state stores information about states of the user accounts;

a smart contract compiler configured to:

receive a smart contract that is associated with the blockchain and the world state data structure;

identify first code in the smart contract that specifies first data and the first Merkle tree;

identify second code in the smart contract that specifies second data and the second Merkle tree; and

compile the smart contract into bytecode; and

a virtual machine that is configured to execute the bytecode to cause the virtual machine to:

store the first data in the first Merkle tree; and

store the second data in the second Merkle tree.

84. — The system of claim 83 in which the compiler is configured to:

recognize code in a smart contract, in which the code identifies a piece of data and a Merkle tree from a plurality of Merkle trees; and

compile the smart contract code into bytecode such that when the bytecode is executed by the virtual machine causes the virtual machine to store the piece of data in the Merkle tree.

85. — The system of claim 83 or 84 in which the compiler is configured to recognize a constant value in the code of the smart contract, and the constant value represents an address of, or a pointer to, the Merkle tree.

86. — The system of claim 83 or 84 in which the compiler is configured to recognize a variable in the code in the smart contract, and the variable represents an address of, or a pointer to, the Merkle tree.

87. — The system of claim 83 or 84 in which the compiler is configured to recognize the code in the smart contract, and the code includes a function for identifying the Merkle tree.

~~88.—The system of claim 83 or 84 in which the compiler is configured to recognize the code in the smart contract, and the code includes a rule set for identifying the Merkle tree.~~

~~89.—The system of any one of claims 83 to 88 in which the first Merkle tree stores state information of user accounts based in a first geographical region, and the second Merkle tree stores state information of user accounts based in a second geographical region.~~

~~90.—The system of any one of claims 83 to 89 in which the smart contract compiler is part of the virtual machine.~~