CLAIMS

What is claimed is:

~~1. A method for executing a smart contract, comprising:~~

~~receiving, by a blockchain node, a request for creating a smart contract, wherein the request comprises bytecode of the smart contract;~~

~~after deploying the smart contract, compiling, by the blockchain node, the bytecode of the smart contract into machine code through Just In Time (JIT) compilation and storing the machine code; and~~

~~interpreting and executing, by the blockchain node when executing the deployed smart contract, the bytecode of the smart contract if the machine code corresponding to the bytecode of the smart contract is not locally stored.~~

~~2. The method according to claim 1, wherein after deploying the smart contract, the compiling, by the blockchain node, the bytecode of the smart contract into machine code through JIT compilation comprises:~~

~~compiling, by the blockchain node, the bytecode of the smart contract through JIT compilation after creating the smart contract; or~~

~~compiling, by the blockchain node, the bytecode of the smart contract through JIT compilation in an idle time period after creating the smart contract; or~~

~~compiling, by the blockchain node, the bytecode of the smart contract through JIT compilation after receiving a transaction request for invoking the created smart contract.~~

~~3. The method according to claim 1, wherein the compiling, by the blockchain node, the bytecode of the smart contract into machine code through JIT compilation and storing the machine code comprises:~~

~~identifying hotspot bytecode for compilation and storage in a process of translating and/or executing the bytecode of the smart contract; and/or~~

~~performing optimized compilation and storing optimized machine code in the process of translating and/or executing the bytecode of the smart contract.~~

~~4. The method according to claim 3, wherein a granularity of identifying the hotspot bytecode comprises a function level or a code block level.~~

~~5. The method according to claim 1, wherein the interpreting and executing, by the blockchain node when executing the deployed smart contract, the bytecode of the smart contract if the machine code of the smart contract is not locally stored comprises:~~

~~when the blockchain node executes the deployed smart contract, if the machine code of the smart contract is not locally stored, further querying, by the blockchain node, a current JIT compilation status of the bytecode of the smart contract, and interpreting and executing the bytecode of the smart contract if the bytecode of the smart contract is being compiled through JIT compilation; or~~

~~when the blockchain node executes the deployed smart contract, if the machine code of the smart contract is not locally stored, further querying, by the blockchain node, a current JIT compilation status of the bytecode of the smart contract, and if JIT compilation has not been started, starting JIT compilation, and interpreting and executing the bytecode of the smart contract.~~

~~6. The method according to claim 1, wherein the interpreting and executing, by the blockchain node, the bytecode of the smart contract comprises:~~

~~interpreting and executing, by the blockchain node, bytecode corresponding to a function to be invoked in the smart contract.~~

~~7. The method according to claim 1, wherein when executing the deployed smart contract, the blockchain node executes the machine code if the machine code of the smart contract is locally stored.~~

~~8. The method according to claim 7, wherein the executing, by the blockchain node, the machine code comprises:~~

~~executing, by the blockchain node, stored machine code corresponding to a function to be invoked and/or a code block in the smart contract.~~

~~9. The method according to any one of claims 1 to 8, wherein the machine code is stored by caching the machine code.~~

1~~0~~. A method for executing a smart contract, comprising:

after deploying bytecode of a smart contract, compiling, by a blockchain node, the bytecode of the smart contract into machine code through Just-In-Time (JIT) compilation and storing the machine code in a memory local to the blockchain node;

determining whether the machine code corresponding to the bytecode of the smart contract to be invoked is locally stored in the memory; and

in response to determining that the machine code corresponding to the bytecode of the smart contract to be invoked is not locally stored in the memory, before the JIT compilation of the bytecode of the smart contract is completed, interpreting and executing, by the blockchain node when executing the ~~deployed~~ smart contract, the bytecode of the smart contract ~~if the machine code corresponding to the bytecode of the~~

~~smart contract is not locally stored~~.

~~11~~2. The method according to claim 1~~0~~, wherein after deploying bytecode of a smart contract, the compiling, by a blockchain node, the bytecode of the smart contract into machine code through JIT compilation comprises:

~~compiling, by the blockchain node, the bytecode of the smart contract through JIT compilation after creating the smart contract; or~~

compiling, by the blockchain node, the bytecode of the smart contract through JIT compilation in an idle time period after creating the smart contract; or

compiling, by the blockchain node, the bytecode of the smart contract through JIT compilation after receiving a transaction request for invoking the created smart contract.

~~12~~3. The method according to claim 1~~0~~, wherein the compiling, by a blockchain node, bytecode of a smart contract into machine code through JIT compilation and storing the machine code comprises:

identifying hotspot bytecode for compilation and storage in a process of translating and/or executing the bytecode of the smart contract; and/or

performing optimized compilation and storing optimized machine code in the process of translating and/or executing the bytecode of the smart contract.

~~13~~4. The method according to claim ~~12~~3, wherein a granularity of identifying the hotspot bytecode comprises a function level or a code block level.

~~14~~5. The method according to claim 1~~0~~, wherein the interpreting and executing, by the blockchain node when executing the ~~deployed~~ smart contract, the bytecode of the smart contract if the machine code of the smart contract is not locally stored comprises:

when the blockchain node executes the ~~deployed~~ smart contract, if the machine code of the smart contract is not locally stored, further querying, by the blockchain node, a current JIT compilation status of the bytecode of the smart contract, and interpreting and executing the bytecode of the smart contract if the bytecode of the smart contract is being compiled through JIT compilation; or

when the blockchain node executes the ~~deployed~~ smart contract, if the machine code of the smart contract is not locally stored, further querying, by the blockchain node, a current JIT compilation status of the bytecode of the smart contract, and if JIT compilation has not been started, starting JIT compilation, and interpreting and executing the bytecode of the smart contract.

~~15~~6. The method according to claim 1~~0~~, wherein the interpreting and executing, by the blockchain node, the bytecode of the smart contract comprises:

interpreting and executing, by the blockchain node, bytecode corresponding to a function to be invoked in the smart contract.

~~16~~7. The method according to claim 1~~0~~, wherein when executing the ~~deployed~~ smart contract, the blockchain node executes the machine code if the machine code of the smart contract is locally stored.

~~17~~8. The method according to claim ~~16~~7, wherein the executing, by the blockchain node, the machine code comprises:

executing, by the blockchain node, stored machine code corresponding to a function to be invoked and/or a code block in the smart contract.

~~18~~9. The method according to any one of claims 1~~0 to 17~~ to 8, wherein the ~~machine code is stored by caching the machine code~~memory is a cache.

1~~0~~9. A blockchain node for executing a smart contract, configured to perform the method according to any one of claims 1 to 9~~ or 10 to 18~~.

~~20~~11. A blockchain node for executing a smart contract, comprising:

a processor; and

a memory, wherein the memory stores a program, and when the processor executes the program, the method according to any one of claims 1 to 9 ~~or 10 to 18~~ is performed.

~~21~~12. A storage medium, configured to store a program, wherein when the program is executed, the operation according to any one of claims 1 to 9 ~~or 10 to 18~~ is performed.