Parallel Proof-of-Work
A cumulative security system for proof-of-work blockchains

Abstract Exogenous mining power presents an ongoing threat to the security of independent blockchains. Two or more independent blockchains implementing a single proof-of-work algorithm will result in only one strongly secure blockchain because no economic incentive prevents mining nodes on the strongest blockchain from attacking a weaker one. Parallel proof-of-work combines the cumulative work done in all participating networks, removing wasted competition, and bringing all networks to consensus by adding up their work.

Introduction

We propose a proof-of-work system separating a block into two distinct logical categories. First, a header containing proof-of-work and a transaction tree, called the *consensus proof*. Second, a set of blocks called *parallel blocks*, each one containing the set of transactions from a corresponding participating blockchain. Nodes compete to find proof-of-work to secure all parallel blocks.

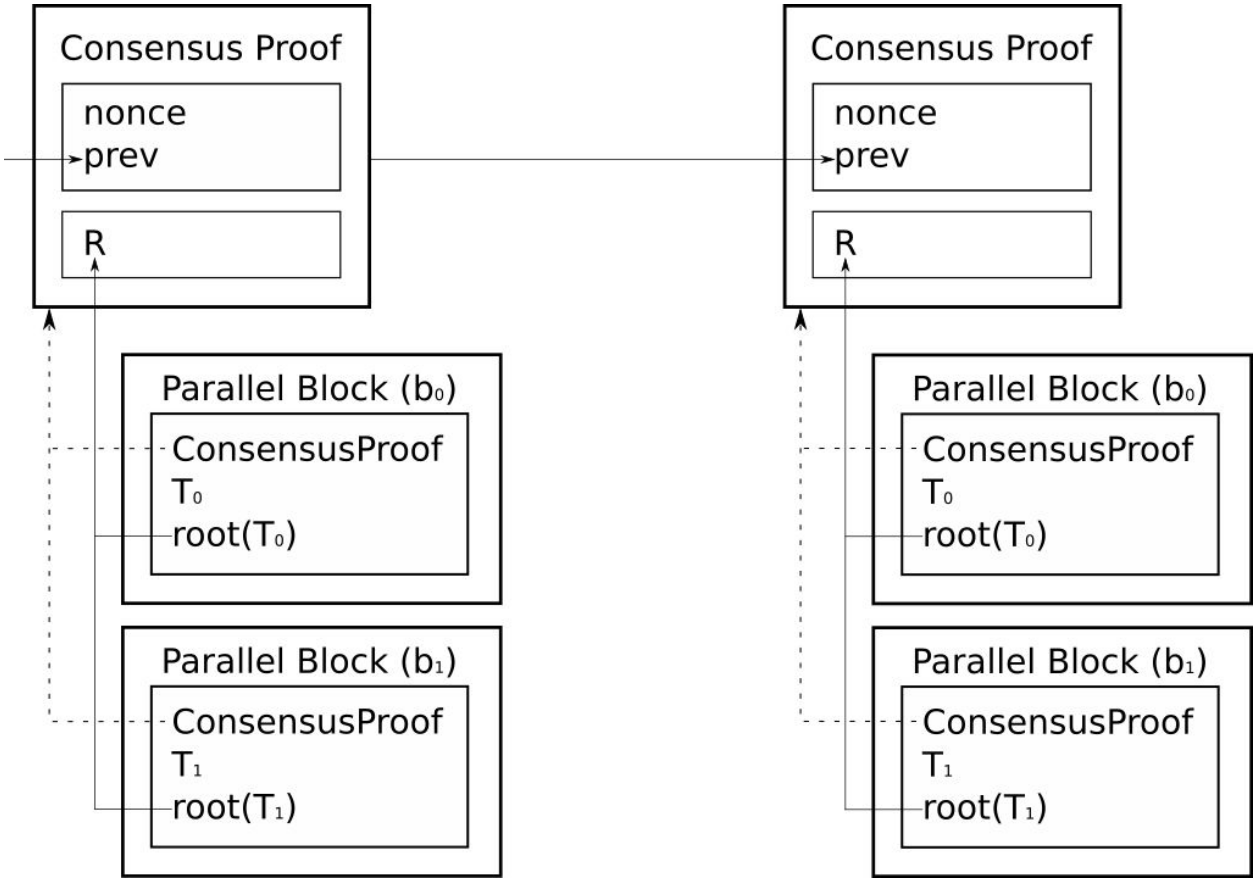
Definitions

b is a *parallel blockchain*, a blockchain participating in the parallel proof-of-work system.
B is the set of all parallel blockchains such that $B = (b_0, b_1, \ldots, b_n)$.
hash(P) is the hash digest of preimage data P.
nonce is the field of data used to satisfy the difficulty requirement for a consensus proof.
prev is the hash of the previous consensus proof.
T_0 is a set of transactions waiting to be mined in a parallel blockchain b_0.
$root(T_0)$ is the merkle root of all transactions within the set of transactions T_0.
R is a merkle root of all merkle roots such that $R = root(root(T_0), root(T_1), \ldots root(T_n))$.

Example

Consider a system with two parallel blockchains b_0 and b_1.

ConsensusProof = hash(nonce, prev, R)
$ParallelBlock(b_0) = hash(ConsensusProof, T_0, root(T_0))$
$ParallelBlock(b_1) = hash(ConsensusProof, T_1, root(T_1))$



The steps to construct the blockchain are mostly the same as classic proof-of-work:

1) New transactions from blockchain b_0 are broadcast to all other b_0 nodes.
2) Nodes attempting to find proof-of-work collect all transactions from all blockchains.
3) When a node finds proof-of-work for a consensus proof, it can issue parallel blocks for each blockchain it has collected transactions for, earning rewards from each.

Result

The result is collaboration between multiple blockchains using the same mining algorithm. Mining nodes that find a valid consensus proof are able to construct parallel blocks for each participating blockchain in the network, generating coins in each blockchain.