Consensus Algorithms in Wireless Blockchain System

1 Consensus Algorithm in Each Round

Algorithm 1 Utilities of Node v

function BlockLeader($seed, w_v, W$) ▷ Choose a Leader
  $(value, proof) = VRF(sk_v, seed)$
  $p_v = \frac{w_v}{W}$
  ID = 0
  if $\frac{value}{2^{bit(value)}} \notin [\sum_{k=0}^{i} p_k, \sum_{k=0}^{i+1} p_k)$ **then**
    $ID++$
  end if
  return ID, value, Proof
end function
function VerifyLeader($pk_v, ID, value, proof, seed, w_v, W$) ▷ Verify the validation of leader
  $result = VerifyVRF(pk_v, value, proof)$
  if $result == 0$ or $\frac{value}{2^{bit(value)}} \notin [\sum_{k=0}^{ID} p_k, \sum_{k=0}^{ID+1} p_k)$ **then**
    return False
  else
    return True
  end if
end function
function GenerateBlock(BC_v, Txs) ▷ Create a new block
  $B_v^{new}.Index \leftarrow Length(BC_v)$
  $B_v^{new}.Prehash \leftarrow BC_v^{latest}.Hash$
  $B_v^{new}.txs \leftarrow Txs$
  $B_v^{new}.Timestamp \leftarrow Timestamp$
  $B_v^{new}.Hash \leftarrow Hash(B_v^{new})$
  return B_v^{new}
end function
function Append(BC_v, B_v^{new}) ▷ Append the new block
  return $BC_v \leftarrow BC_v + B_v^{new}$
end function
function ValidBlock(BC_v, B_v^{new}) ▷ verify the validation of new block
  if $B_v^{new}.Prehash! = BC_v^{latest}.Hash$ **then**
    return False
  else if $ValidTansactions(B_v^{new}.txs) == False$ **then**
    return False
  else if $VerifyLeader(pk_v, ID, value, proof, seed, w_v, W) == False$ **then**
    return False
  else
    return True
  end if
end function

function VALIDTRANSACTIONS(Txs)
  for $tx \in Txs$ **do**
    if tx is Invalid **then**
      return False
    end if
  end for
  return True
end function
function MSGB($BC_v, B_v^{new}, w_v, ID, value, proof$)
  $m_B.latest \leftarrow BC_v^{latest}$
  $m_B.B_v^{new} \leftarrow B_v^{new}$
  $m_B.sortition \leftarrow \{w_v, ID, value, proof\}$
  return m_B
end function

Algorithm 2 Consensus Algorithm

▷ Initialization:
▷ Leader Election and Block Propocal:
$(BL, value, Proof) = BlockLeader(seed)$
if $v == BL$ **then**
  $B_v^{new} = GenerateBlock(BC_v, Txs)$
  $m_B = MSGB(BC_v, B_v^{new}, w_v, value, proof)$
  $broadcast(m_B, \sigma_v)$
end if
▷ Block Verification and Finalization:
while $!Confirmed$ **do**
  $(sigShares, FinalSig, B_v^{new}) = RecNewBlock(m_B, \sigma_v)$
  if $isValid(FinalSig)$ **then**
    $\sigma_F = FinalSig$
    $broadcast(m_B, \sigma_F)$
    $Append(BC_v, B_v^{new})$
    $Confirmed = True$
  else if $Count(SigShares) > K$ **then**
    $\sigma_F = RecoverFinalSig(sigShare)$
    $broadcast(m_B, \sigma_F)$
    $Append(BC_v, B_v^{new})$
    $Confirmed = True$
  else if $\sigma_v \notin sigShares$ **then**
    $sigShares = AppendSignature(\sigma_v)$
    $broadcast(m_B, sigShares)$
  else
    $broadcast(m_B, sigShares)$
  end if
end while
function RECNEWBLOCK(m_B, σ_v)
  if $\sigma_v \notin sigShares$ **then**
    $sigShares = AppendSignature(\sigma_v)$
  end if
  if $Count(sigShares) > K$ **then**
    $FinalSig = RecoverFinalSig(sigShares)$
  else

36: $FinalSig = null$
37: **end if**
38: **return** $sigShares, FinalSig, B_v^{new}$
39: **end function**
40: **function** AppendSignature(σ_v)
41: **if** $\sigma_v \notin sigShares$ **then**
42: $sigShares \leftarrow sigShares + \sigma_v)$
43: **end if**
44: **return** $sigShares$
45: **end function**