Algorithm 1: Handling Inconsistent Blockchain Views (G_0, B_0)

Input: Blockchain graph G_0, genesis block B_0
Output: Target close-sibling s

// Collecting all whole-siblings to a set S_0
$S_0 \leftarrow \varnothing$
foreach $B \in G_0$ **do**
  $s \leftarrow$ BlocktoWholeSibling(B)
  if $s \in S_0$ **then**
    continue
  else
    $S_0 \leftarrow S_0 \cup \{s\}$
  end
end
// Initialize weights of each whole-sibling as 0
foreach $s \in S_0$ **do**
  $W(s) \leftarrow 0$
end
// Calculate weights of all whole-siblings
foreach $B \in G_0$ **do**
  foreach $s \in S_0$ **do**
    foreach $v \in$ *SubviewsBeginWith*(s) **do**
      if $B \in v$ **then**
        $W(s) \leftarrow W(s) +$ Miningpower(B)
        break
      end
    end
  end
end
// Find the target whole-sibling
$s \leftarrow$ BlocktoWholeSibling(B_0)
if *ChildWholeSiblings*$(s) = \varnothing$ **then**
  return s
else
  Update $s \leftarrow \underset{s' \in \text{ChildWholeSiblings}(s)}{\arg\max} W(s')$
  Go to line 24
end

Function: Subviews Begin with

```
// Function to find all subviews that begin with s
Function SubviewsBeginWith(s):
    V ← ∅
    if ChildWholeSiblings(s) = ∅ then
        v ← WholeSiblingstoSubview(s)
        V ← V.Append(v)
        return V
    else
        foreach c ∈ ChildWholeSiblings(s) do
            foreach v ∈ SubviewsBeginWith(c) do
                V.Append(v.Append(s))
            end
        end
    end
return V
```

Function: Child Whole Siblings

```
// Function to find child whole siblings
Function ChildWholeSiblings(s):
    C ← ∅
    foreach s′ ∈ S_0 do
        foreach B′ ∈ s′ do
            foreach B ∈ s do
                if B ∈ B′.predecessors then
                    C.Append(s′)
                end
            end
        end
    end
return C
```