
MODULE *BlockDag*

In this specification we define notions on *DAGs* useful for DAG-based consensus protocols (which build *DAGs* of blocks)

EXTENDS *FiniteSets*, *Sequences*, *Integers*, *Utils*, *Digraph*, *TLC*

CONSTANTS

N The set of all network nodes (not *DAG* nodes)
 R The set of rounds
 $\text{Leader}(_)$ operator mapping each round to its leader

For our purpose of checking safety and liveness, *DAG* vertices just consist of a node and a round.

$V \triangleq N \times R$ the set of possible *DAG* vertices
 $\text{Node}(v) \triangleq v[1]$
 $\text{Round}(v) \triangleq \text{IF } v = \langle \rangle \text{ THEN } 0 \text{ ELSE } v[2]$ accomodates $\langle \rangle$ as default value

Next we define leader vertices:

$\text{LeaderVertex}(r) \triangleq \text{IF } r > 0 \text{ THEN } \langle \text{Leader}(r), r \rangle \text{ ELSE } \langle \rangle$
 $\text{IsLeader}(v) \triangleq \text{LeaderVertex}(\text{Round}(v)) = v$
 $\text{Genesis} \triangleq \langle \rangle$
ASSUME $\text{IsLeader}(\text{Genesis})$ this should hold

$\text{OrderSet}(S)$ arbitrarily order the members of the set S . Note that, in TLA+, `CHOOSE` is deterministic but arbitrary choice, *i.e.* `CHOOSE $e \in S : \text{TRUE}$` is always the same e if S is the same

RECURSIVE $\text{OrderSet}(_)$
 $\text{OrderSet}(S) \triangleq \text{IF } S = \{\} \text{ THEN } \langle \rangle \text{ ELSE}$
LET $e \triangleq \text{CHOOSE } e \in S : \text{TRUE}$
IN $\text{Append}(\text{OrderSet}(S \setminus \{e\}), e)$

$\text{PreviousLeader}(\text{dag}, r)$ returns the leader vertex in dag whose round is the largest but smaller than r . We assume that dag contains at least the genesis vertex.

$\text{PreviousLeader}(\text{dag}, r) \triangleq \text{CHOOSE } l \in \text{Vertices}(\text{dag}) :$
 $\wedge \text{IsLeader}(l)$
 $\wedge \text{Round}(l) = \text{Max}(\{\text{Round}(l2) : l2 \in$
 $\{l2 \in \text{Vertices}(\text{dag}) : \text{IsLeader}(l2) \wedge \text{Round}(l2) < r\}\})$

Linearize a *DAG*. In a real blockchain we should use a topological ordering, but, for the purpose of ensuring agreement, an arbitrary ordering (as provided by OrderSet) is fine. This assume a *DAG* where all paths end with the *Genesis* vertex.

RECURSIVE $\text{Linearize}(_, _)$
 $\text{Linearize}(\text{dag}, l) \triangleq \text{IF } \text{Vertices}(\text{dag}) = \{\langle \rangle\} \text{ THEN } \langle \rangle \text{ ELSE}$
LET $\text{dagOfL} \triangleq \text{SubDag}(\text{dag}, \{l\})$
 $\text{prevL} \triangleq \text{PreviousLeader}(\text{dagOfL}, \text{Round}(l))$
 $\text{dagOfPrev} \triangleq \text{SubDag}(\text{dag}, \{\text{prevL}\})$
 $\text{remaining} \triangleq \text{Vertices}(\text{dagOfL}) \setminus \text{Vertices}(\text{dagOfPrev})$

IN $\text{Linearize}(\text{dagOfPrev}, \text{prevL}) \circ \text{OrderSet}(\text{remaining} \setminus \{l\}) \circ \langle l \rangle$

$\text{Compatible}(s1, s2) \triangleq$ whether the sequence $s1$ is a prefix of the sequence $s2$, or vice versa
 $\forall i \in 1 \dots \text{Min}(\{\text{Len}(s1), \text{Len}(s2)\}) : s1[i] = s2[i]$