```
──────────────────────────── MODULE CC ────────────────────────────
TLA+ specification of Causal Consistency variants, including CC, CM, and CCv.

See the paper “On Verifying Causal Consistency“ (POPL’2017).

EXTENDS Naturals, Sequences, FiniteSets, Functions, FiniteSetsExt,
        RelationUtils, TLC

CONSTANTS Keys, Vals
InitVal ≜ 0   we follow the convention in POPL’2017

 oid: unique operation identifier
Operation ≜ [type : {“read”, “write”}, key : Keys, val : Vals, oid : Nat]
R(k, v, oid)  ≜ [type ↦ “read”, key ↦ k, val ↦ v, oid ↦ oid]
W(k, v, oid)  ≜ [type ↦ “write”, key ↦ k, val ↦ v, oid ↦ oid]

Session ≜ Seq(Operation)   A session s ∈ Session is a sequence of operations.
History ≜ SUBSET Session   A history h ∈ History is a set of sessions.
───────────────────────────────────────────────────────────────────
Utilities.

Ops(h) ≜   Return the set of all operations in history h ∈ History.
    UNION {Range(s) : s ∈ h}

 ReadOps(h) ≜ \* Return the set of all read operations in history h ∈ History.
   {op ∈ Ops(h) : op.type = ”read“}

 WriteOps(h) ≜ \* Return the set of all write operations in history h ∈ History.
   {op ∈ Ops(h) : op.type = ”write“}
───────────────────────────────────────────────────────────────────
Well-formedness of history h ∈ History:

- TODO: type invariants
- uniqueness of oids
WellFormed(h) ≜
   ∧ h ∈ History
   ∧ Cardinality(Ops(h)) = ReduceSet(LAMBDA s, x : Len(s) + x, h, 0)
───────────────────────────────────────────────────────────────────
Auxiliary definitions for the axioms used in the definitions of causal consistency
 The program order of h ∈ History is a union of total orders among operations in the same session
ProgramOrder(h) ≜ UNION {Seq2Rel(s) : s ∈ h}

 The set of operations that preceed o ∈ Operation in program order in history h ∈ History
POPast(h, o) ≜ InverseImage(ProgramOrder(h), o)

 The set of operations that preceed o ∈ Operation in causal order co
CausalPast(co, o) ≜ InverseImage(co, o)

 The restriction of causal order co to the operations in the causal past of operation o ∈ Operation
CausalHist(co, o) ≜ co | CausalPast(co, o)
```

```
The restriction of arbitration $arb$ to the operations in the causal past of operation $o \in Operation$
$CausalArb(co, arb, o) \triangleq arb \mid CausalPast(co, o)$
------------------------------------------------------------------------
Axioms used in the defintions of causal consistency

$RWRegSemantics(seq, o) \triangleq$  Is $o \in Operation$ legal when it is appended to $seq$
    IF $o.type =$ "write" THEN TRUE
     ELSE LET $wseq \triangleq SelectSeq(seq,$ LAMBDA $op : op.type =$ "write" $\land\ op.key = o.key)$
            IN  IF $wseq = \langle\rangle$ THEN $o.val = InitVal$
                  ELSE  $o.val = wseq[Len(wseq)].val$

$AxCausalValue(co, o) \triangleq$
    LET $seqs \triangleq AllLinearExtensions(CausalHist(co, o), CausalPast(co, o))$
    IN  TRUE $\in \{RWRegSemantics(seq, o) : seq \in seqs\}$  TODO: shortcut implementation of anyTrue for efficiency

$AxCausalArb(co, arb, o) \triangleq$
    LET $seq \triangleq AnyLinearExtension(CausalArb(co, arb, o), CausalPast(co, o))$ it is unique
    IN  $RWRegSemantics(seq, o)$
------------------------------------------------------------------------
Specification of CC

$CC(h) \triangleq$  Check whether $h \in History$ satisfies CC (Causal Consistency)
      LET $ops \triangleq Ops(h)$
      IN  $\exists\, co \in$ SUBSET $(ops \times ops)$ :  TODO: to generate (given a chain decomposition)
              $\land\ Respect(co, ProgramOrder(h))$                    AxCausal
              $\land\ IsStrictPartialOrder(co, ops)$
              $\land\ PrintT($"co: " $\circ\ ToString(co))$
              $\land\ \forall\, o \in ops : AxCausalValue(co, o)$             AxCausalValue
------------------------------------------------------------------------
Specification of CCv


To generate possible ordering relations, not to enumerate and test them
$CCv(h) \triangleq$  Check whether $h \in History$ satisfies CCv (Causal Convergence)
      LET $ops \triangleq Ops(h)$
      IN  $\exists\, co \in$ SUBSET $(ops \times ops)$ :  TODO: to generate (given a chain decomposition)
              $\land\ Respect(co, ProgramOrder(h))$                    AxCausal
              $\land\ IsStrictPartialOrder(co, ops)$
              $\land\ PrintT($"co: " $\circ\ ToString(co))$
              $\land\ \exists\, arb \in \{Seq2Rel(le) : le \in AllLinearExtensions(co, ops)\}$ :  AxArb
                  $\land\ \forall\, o \in ops : AxCausalArb(co, arb, o)$  AxCausalArb
                  $\land\ PrintT($"arb: " $\circ\ ToString(arb))$
Version 2: re-arrange clauses
$CCv2(h) \triangleq$  Check whether $h \in History$ satisfies CCv (Causal Convergence)
      LET $ops \triangleq Ops(h)$
      IN  $\exists\, co \in$ SUBSET $(ops \times ops)$ :  FIXME: efficiency!!!
              $\land\ Respect(co, ProgramOrder(h))$  AxCausal
```

```
                ∧ IsStrictPartialOrder(co, ops)
                ∧ PrintT("co: " ∘ ToString(co))
                ∧ ∃ arb ∈ SUBSET (ops × ops) :   to generate; not to test
                      ∧ Respect(arb, co)                              AxArb
                      ∧ IsStrictTotalOrder(arb, ops)
                      ∧ ∀ o ∈ ops : AxCausalArb(co, arb, o)   AxCausalArb
                      ∧ PrintT("arb: " ∘ ToString(arb))

 Version 1: Following the definition of POPL2017

CCv1(h) ≜   Check whether h ∈ History satisfies CCv (Causal Convergence)
      LET ops ≜ Ops(h)
      IN  ∃ co ∈ SUBSET (ops × ops) :   FIXME: efficiency!!!
                ∧ ∃ arb ∈ SUBSET (ops × ops) :
                    ∧ PrintT("co: " ∘ ToString(co))
                    ∧ PrintT("arb: " ∘ ToString(arb))
                    ∧ IsStrictPartialOrder(co, ops)
                    ∧ IsStrictTotalOrder(arb, ops)
                    ∧ Respect(co, ProgramOrder(h))               AxCausal
                    ∧ Respect(arb, co)                           AxArb
                    ∧ ∀ o ∈ ops : AxCausalArb(co, arb, o)        AxCausalArb

\* Modification History
\* Last modified Sun Apr 18 10:31:01 CST 2021 by hengxin
\* Created Tue Apr 01 10:24:07 CST 2021 by hengxin
```