MODULE *CCTest*

Test of *CC* Module

EXTENDS *CC*

Test case: The following histories are from Figure 2 of the *POPL*'2017 paper.

Naming Conventions:

– *ha*: history of Figure 2(*a*)
– *hasa*: session a of history ha

TODO:

- to add more test cases
- to automatically generate test cases that do or do not satisfy the specs

- consider Section 3.2 of *POPL*'2017
– *ref*: the *MonkeyDB* paper

$hasa \triangleq \langle W(\text{“x”}, 1, 1), R(\text{“x”}, 2, 2)\rangle$
$hasb \triangleq \langle W(\text{“x”}, 2, 3), R(\text{“x”}, 1, 4)\rangle$
$ha \triangleq \{hasa, hasb\}$ *CM* but not *CCv*

$hbsa \triangleq \langle W(\text{“z”}, 1, 1), W(\text{“x”}, 1, 2), W(\text{“y”}, 1, 3)\rangle$
$hbsb \triangleq \langle W(\text{“x”}, 2, 4), R(\text{“z”}, 0, 5), R(\text{“y”}, 1, 6), R(\text{“x”}, 2, 7)\rangle$
$hb \triangleq \{hbsa, hbsb\}$ *CCv* but not *CM*

$hcsa \triangleq \langle W(\text{“x”}, 1, 1)\rangle$
$hcsb \triangleq \langle W(\text{“x”}, 2, 2), R(\text{“x”}, 1, 3), R(\text{“x”}, 2, 4)\rangle$
$hc \triangleq \{hcsa, hcsb\}$ *CC* but not *CM* nor *CCv*

$hdsa \triangleq \langle W(\text{”}x\text{“}, 1, 1), R(\text{”}y\text{“}, 0, 2), W(\text{”}y\text{“}, 1, 3), R(\text{”}x\text{“}, 1, 4)\rangle$
$hdsb \triangleq \langle W(\text{”}x\text{“}, 2, 5), R(\text{”}y\text{“}, 0, 6), W(\text{”}y\text{“}, 2, 7), R(\text{”}x\text{“}, 2, 8)\rangle$
$hd \triangleq \{hdsa, hdsb\} \setminus^*$ *CC*, *CM*, and *CCv* but no *SC*

$hdsa \triangleq \langle W(\text{“x”}, 1, 1), W(\text{“y”}, 2, 2), R(\text{“y”}, 2, 3)\rangle$
$hdsb \triangleq \langle W(\text{“y”}, 1, 4), R(\text{“x”}, 1, 5), R(\text{“y”}, 1, 6)\rangle$
$hd \triangleq \{hdsa, hdsb\}$ *CC*, *CM*, and *CCv* but no *SC*

$hesa \triangleq \langle W(\text{“x”}, 1, 1), W(\text{“y”}, 1, 2)\rangle$
$hesb \triangleq \langle R(\text{“y”}, 1, 3), W(\text{“x”}, 2, 4)\rangle$
$hesc \triangleq \langle R(\text{“x”}, 2, 5), R(\text{“x”}, 1, 6)\rangle$
$he \triangleq \{hesa, hesb, hesc\}$ not *CC* (nor *CM*, nor *CCv*)

$all \triangleq \{ha, hb, hc, hd, he\}$
$satCC \triangleq \{ha, hb, hc, hd\}$
$satCM \triangleq \{ha, hd\}$
$satCCv \triangleq \{hb, hd\}$

$all \triangleq \{ha, hc, hd, he\}$
$satCC \triangleq \{ha, hc, hd\}$

```
satCM  ≜ {ha, hd}
satCCv ≜ {hd}

----------------------------------------------------------------
WellFormedTest ≜
    ∀ h ∈ all : WellFormed(h)
----------------------------------------------------------------
  Test the self-defined EnumerateRO
EasyPO(s) ≜
    LET rels ≜ SUBSET (s × s)
    IN  {po ∈ rels : IsStrictPartialOrder(po, s)}

EnumeratePOTest ≜
  LET pos ≜ partialOrderSubset({0, 1})
    LET ops ≜ {W("x", 2, 0), R("x", 1, 1), R("x", 1, 2)}
        pos1 ≜ EasyPO(ops)
        pos2 ≜ StrictPartialOrderSubset(ops)
    IN  ∧ pos1 = pos2
        ∧ ∀ po ∈ pos1 :
            PrintT("po: " ∘ ToString(po))
----------------------------------------------------------------
  Test of utility operators for operations
OpsTest ≜
    ∧ PrintT("OpsTest Begin")
     on history ha
    ∧ Ops(ha) = {W("x", 1, 1), R("x", 2, 2), W("x", 2, 3), R("x", 1, 4)}
    ∧ ReadOps(ha) = {R("x", 2, 2), R("x", 1, 4)}
    ∧ ReadOpsOnKey(ha, "x") = {R("x", 2, 2), R("x", 1, 4)}
    ∧ WriteOps(ha) = {W("x", 1, 1), W("x", 2, 3)}
    ∧ WriteOpsOnKey(ha, "x") = {W("x", 1, 1), W("x", 2, 3)}
     on history he
    ∧ Ops(he) = {W("x", 1, 1), W("y", 1, 2), R("y", 1, 3), W("x", 2, 4), R("x", 2, 5), R("x", 1, 6)}
    ∧ ReadOps(he) = {R("y", 1, 3), R("x", 2, 5), R("x", 1, 6)}
    ∧ ReadOpsOnKey(he, "x") = {R("x", 2, 5), R("x", 1, 6)}
    ∧ WriteOps(he) = {W("x", 1, 1), W("y", 1, 2), W("x", 2, 4)}
    ∧ WriteOpsOnKey(he, "y") = {W("y", 1, 2)}
    ∧ PrintT("OpsTest End")
----------------------------------------------------------------
  Test of the auxiliary definitions for the axioms
CardOfProgramOrderOfHistory(h) ≜
    LET CardOfProgramOrderOfSession(s) ≜
          IF Len(s) ≤ 1 THEN 0 ELSE Sum(1 .. Len(s) − 1)
    IN    ReduceSet(LAMBDA s, x : CardOfProgramOrderOfSession(s) + x, h, 0)

THEOREM ProgramOrderCardinalityTheorem ≜  test of PO(h)
```

```
    ∀ h ∈ {ha, hb, hc, hd, he} :
        Cardinality(PO(h)) = CardOfProgramOrderOfHistory(h)

POPastTest ≜  test of POPast(h, o)
    ∧ PrintT("POPastTest Begin")
    ∧ POPast(ha, R("x", 2, 2))  = {W("x", 1, 1)}
    ∧ POPast(hb, R("y", 1, 6))  = {W("x", 2, 4), R("z", 0, 5)}
    ∧ POPast(hc, W("x", 2, 2)) = {}
    ∧ POPast(hd, R("x", 1, 4))  = {W("x", 1, 1), R("y", 0, 2), W("y", 1, 3)}
    ∧ POPast(he, W("x", 2, 4)) = {R("y", 1, 3)}
    ∧ PrintT("POPastTest End")

CausalPastTest ≜  TODO: test of CausalPast(co, o)
    ∧ PrintT("CausalPastTest Begin")
    ∧ LET co ≜ CO(ha)
          o ≜ R("x", 2, 2)
      IN  CausalPast(co, o) = {W("x", 1, 1), R("x", 2, 2), W("x", 2, 3)}
    ∧ PrintT("CausalPastTest End")

CausalHistTest ≜  TODO: test of CausalHist(co, o)
    ∧ PrintT("CausalHistTest Begin")
    ∧ LET co ≜ CO(ha)
          o ≜ R("x", 2, 2)
      IN  CausalHist(co, o) = {⟨W("x", 1, 1), R("x", 2, 2)⟩, ⟨W("x", 2, 3), R("x", 2, 2)⟩}
    ∧ PrintT("CausalHistTest End")

CausalArbTest ≜  TODO: test of CausalArb(co, ar, o)
    ∧ PrintT("CausalArbTest Begin")
    ∧ FALSE
    ∧ PrintT("CausalArbTest End")

AuxiliaryTest ≜  test the auxiliary
    ∧ POPastTest
    ∧ CausalPastTest
    ∧ CausalHistTest
     ∧ CausalArbTest

├──────────────────────────────────────────────────────────────────────────────┤
  Test of axioms

RWRegSemanticsTest ≜  test of RWRegSemanticsTest(seq, o)
    ∧ PrintT("RWRegSemanticsTest Begin")
     seq = ⟨⟩
    ∧ RWRegSemantics(⟨⟩, R("x", InitVal, 1))
    ∧ RWRegSemantics(⟨⟩, W("x", 1, 1))
    ∧ ¬RWRegSemantics(⟨⟩, R("x", 2, 1))
     no W("x", _, _) in seq
    ∧ RWRegSemantics(⟨W("y", 1, 1), W("z", 1, 2), W("y", 1, 3)⟩, R("x", InitVal, 4))
```

```
156     ∧ RWRegSemantics(⟨W("y", 1, 1), W("z", 1, 2), W("y", 1, 3)⟩, W("x", 1, 4))
157     ∧ ¬RWRegSemantics(⟨W("y", 1, 1), W("z", 1, 2), W("y", 1, 3)⟩, R("x", 1, 4))
158      contains W("x", _, _) in seq
159     ∧ RWRegSemantics(⟨W("x", 1, 1), W("y", 1, 2), W("x", 2, 3), W("z", 1, 4)⟩, R("x", 2, 5))
160     ∧ ¬RWRegSemantics(⟨W("x", 1, 1), W("y", 1, 2), W("x", 2, 3), W("z", 1, 4)⟩, R("x", 1, 5))
161     ∧ PrintT("RWRegSemanticsTest End")

163 AxCausalValueTest ≜  TODO: test of AxCausalValue()
164     ∧ PrintT("AxCausalValueTest Begin")
165     ∧ LET  co  ≜ CO(ha)
166            o1  ≜ W("x", 1, 1)
167            o2  ≜ R("x", 2, 2)
168            o3  ≜ W("x", 2, 3)
169            o4  ≜ R("x", 1, 4)
170       IN    ∧ AxCausalValue(co, o1)
171             ∧ AxCausalValue(co, o2)
172             ∧ AxCausalValue(co, o3)
173             ∧ AxCausalValue(co, o4)
174     ∧ PrintT("AxCausalValueTest End")

176 AxCausalSeqTest ≜  Test of AxCausalSeq
177     ∧ PrintT("AxCausalSeqTest Begin")
178     ∧ LET  co  ≜ CO(ha)
179            o1  ≜ W("x", 1, 1)
180            o2  ≜ R("x", 2, 2)
181            o3  ≜ W("x", 2, 3)
182            o4  ≜ R("x", 1, 4)
183       IN    ∧ AxCausalSeq(ha, co, o1)
184             ∧ AxCausalSeq(ha, co, o2)
185             ∧ AxCausalSeq(ha, co, o3)
186             ∧ AxCausalSeq(ha, co, o4)
187     ∧ PrintT("AxCausalSeqTest End")


190 AxCausalArbTest ≜  TODO: test of AxCausalArb()
191     ∧ PrintT("AxCausalArbTest Begin")
192     ∧ FALSE
193     ∧ PrintT("AxCausalArbTest End")

195 Axioms ≜  Test the axioms
196     ∧  RWRegSemanticsTest
197     ∧  AxCausalValueTest
198     ∧  AxCausalSeqTest
199      ∧ AxCausalArbTest
200 ├──────────────────────────────────────────────────────────────────────────┤
    Test of the relations defined for bad patterns
```

```
oidHB(h) ≜  All happened-before relation for o ∈ history h repsented by oid
    LET oidHBo(o) ≜ {⟨o1.oid − 1, o2.oid − 1⟩ : ⟨o1, o2⟩ ∈ HBo(h, o)}
    IN  {⟨o.oid − 1, oidHBo(o)⟩ : o ∈ Ops(h)}

HBTest ≜
    ∧ PrintT("HBTest Begin")
    ∧ PrintT(oidHB(ha))
    ∧ PrintT(oidHB(hb))
    ∧ PrintT(oidHB(hc))
    ∧ PrintT(oidHB(hd))
    ∧ PrintT(oidHB(he))
    ∧ PrintT("HBTest End")
```

Test of the definitions of causal consistency

ha: 4; *hb*: 7; *hc*: 4; *hd*: 8; he: 6

```
CCDefTest ≜
    ∧ PrintT("CCDefTest Begin")
    ∧ ∀ h ∈ satCC :
        ∧ PrintT(h)
        ∧ CC(h)
    ∧ ∀ h ∈ all \ satCC :
        ∧ PrintT(h)
        ∧ ¬CC(h)
    ∧ PrintT("CCDefTest End")

CCvDefTest ≜
    ∧ PrintT("CCvDefTest Begin")
    ∧ ∀ h ∈ satCCv :
        ∧ PrintT(h)
        ∧ CCv(h)
    ∧ ∀ h ∈ all \ satCCv :
        ∧ PrintT(h)
        ∧ ¬CCv(h)
    ∧ PrintT("CCvDefTest End")

CMDefTest ≜
    ∧ PrintT("CMDefTest Begin")
    ∧ ∀ h ∈ satCM :
        ∧ PrintT(h)
        ∧ CM(h)
    ∧ ∀ h ∈ all \ satCM :
        ∧ PrintT(h)
        ∧ ¬CM(h)
    ∧ PrintT("CMDefTest End")
```

```
CausalDefTest ≜
    ∧ CCDefTest
    ∧ CCvDefTest
    ∧ CMDefTest
```

Test of the checking algorithms for causal consistency

ha: 4; *hb*: 7; *hc*: 4; *hd*: 8; he: 6

```
CCAlgTest ≜ Test of the checking algorithm CCAlg for CC (Causal Consistency)
    LET sat ≜ satCC
    IN  ∧ ∀ h ∈ sat :
            ∧ PrintT(ToString(h) ∘ " is differentiated: " ∘ ToString(IsDifferentiated(h)))
            ∧ CCAlg(h)
        ∧ ∀ h ∈ all \ sat :
            ∧ PrintT(ToString(h) ∘ " is differentiated: " ∘ ToString(IsDifferentiated(h)))
            ∧ ¬CCAlg(h)

CCvAlgTest ≜ Test of the checking algorithm CCvAlg for CCv (Causal Convergence)
    LET sat ≜ satCCv
    IN  ∧ ∀ h ∈ sat :
            ∧ PrintT(ToString(h) ∘ " is differentiated: " ∘ ToString(IsDifferentiated(h)))
            ∧ CCvAlg(h)
        ∧ ∀ h ∈ all \ sat :
            ∧ PrintT(ToString(h) ∘ " is differentiated: " ∘ ToString(IsDifferentiated(h)))
            ∧ ¬CCvAlg(h)

CMAlgTest ≜ Test of the checking algorithm CMAlg for CM (Causal Memory)
    LET sat ≜ satCM
    IN  ∧ ∀ h ∈ sat :
            ∧ PrintT(ToString(h) ∘ " is differentiated: " ∘ ToString(IsDifferentiated(h)))
            ∧ CMAlg(h)
        ∧ ∀ h ∈ all \ sat :
            ∧ PrintT(ToString(h) ∘ " is differentiated: " ∘ ToString(IsDifferentiated(h)))
            ∧ ¬CMAlg(h)

CausalAlgTest ≜
    ∧ CCAlgTest
    ∧ CCvAlgTest
    ∧ CMAlgTest
```

```
VARIABLES x  keep it so that the model can be run
```

* Modification *History*
* Last modified *Fri* May 28 11:05:56 *CST* 2021 by Young
* Last modified *Thu Apr* 22 15:12:59 *CST* 2021 by *hengxin*
* Created *Fri Apr* 09 11:53:33 *CST* 2021 by *hengxin*