```
1  ---------- MODULE StateSpace ----------
   The graph representation of n-ary ordered state space and 2D state space used in CJupiter and
   XJupiter, respectively.
6  EXTENDS JupiterCtx, GraphsUtil
7  ----------
8  IsSS(G) ≜  A state space is a digraph with labeled edges.
9        ∧ IsGraph(G)  It is a digraph (represented by a record).
10       ∧ G.node ⊆ (SUBSET Oid)  Each node represents a document state, i.e., a set of Oid.
11       ∧ G.edge ⊆ [from : G.node, to : G.node, cop : Cop]  Each edge is labeled with a Cop operation.

13 EmptySS ≜ EmptyGraph
14 ----------
15 Locate(cop, ss) ≜  Locate the node in state space ss that matches the context of cop.
16     CHOOSE n ∈ ss.node : n = cop.ctx

18 xForm(NextEdge(_, _, _), r, cop, ss) ≜  Transform cop with an operation sequence
19     LET u ≜ Locate(cop, ss)                in state space ss at replica r.
20         v ≜ u ∪ {cop.oid}
21         RECURSIVE xFormHelper(_, _, _, _)
22          xFormHelper(uh, vh, coph, xss) ≜   xss: eXtra ss created during transformation
23              IF uh = ds[r] THEN [xcop ↦ coph,
24                                  xss ↦ xss,
25                                  lss ↦ [node ↦ {vh},
26                                         edge ↦ {[from ↦ uh, to ↦ vh, cop ↦ coph]}]]
27               ELSE LET e ≜ NextEdge(r, uh, ss)
28                        copprime ≜ e.cop
29                        uprime ≜ e.to
30                        vprime ≜ vh ∪ {copprime.oid}
31                         coph2copprime ≜ COT(coph, copprime)
32                         copprime2coph ≜ COT(copprime, coph)
33                    IN   xFormHelper(uprime, vprime, coph2copprime,
34                            xss ⊕ [node ↦ {vprime},
35                                   edge ↦ {[from ↦ vh, to ↦ vprime,
36                                            cop ↦ copprime2coph],
37                                           [from ↦ uprime, to ↦ vprime,
38                                            cop ↦ coph2copprime]}])
39     IN   xFormHelper(u, v, cop, [node ↦ {v},
40                                  edge ↦ {[from ↦ u, to ↦ v, cop ↦ cop]}])

42 RECURSIVE ExtractCopSeq(_, _, _, _) \* Extract a Cop sequence starting with u in ss at replica r.
43 ExtractCopSeq(NextEdge(_, _, _), r, u, ss) ≜
44   IF u = ds[r] THEN ⟨⟩
45    ELSE LET e ≜ NextEdge(r, u, ss)
46       IN ⟨e.cop⟩ ∘ ExtractCopSeq(NextEdge, r, e.to, ss)

48 xFormCopCopsSS(cop, cops) ≜ \* Transform cop against cops (a sequence of Cop) on state space.
```

```
49    LET RECURSIVE xFormCopCopsSSHelper(_, _, _)
50      xFormCopCopsSSHelper(coph, copsh, xss) ≜
51        LET u ≜ coph.ctx
52            v ≜ u ∪ {coph.oid}
53          uvSS ≜ [node ↦ {u, v}, edge ↦ {[from ↦ u, to ↦ v, cop ↦ coph]}]
54          IN  IF copsh = ⟨⟩ THEN [xcop ↦ coph, xss ↦ xss ⊕ uvSS, lss ↦ uvSS]
55              ELSE LET copprimeh ≜ Head(copsh)
56                       uprime ≜ u ∪ {copprimeh.oid}
57                       vprime ≜ u ∪ {coph.oid, copprimeh.oid}
58                   coph2copprimeh ≜ COT(coph, copprimeh)
59                   copprimeh2coph ≜ COT(copprimeh, coph)
60                IN  xFormCopCopsSSHelper(coph2copprimeh, Tail(copsh),
61                       xss ⊕ [node ↦ {u, v},
62                              edge ↦ {[from ↦ u, to ↦ v, cop ↦ coph],
63                                      [from ↦ u, to ↦ uprime, cop ↦ copprimeh],
64                                      [from ↦ v, to ↦ vprime, cop ↦ copprimeh2coph]}])
65    IN  xFormCopCopsSSHelper(cop, cops, EmptySS)
66
67  xFormSS(cop, copprime) ≜  \* Transform cop against copprime on state space.
68    LET u ≜ cop.ctx      \* Return the extra state space.
69        v ≜ u ∪ {cop.oid}
70        uprime ≜ u ∪ {copprime.oid}
71        vprime ≜ u ∪ {cop.oid, copprime.oid}
72        cop2copprime ≜ COT(cop, copprime)
73        copprime2cop ≜ COT(copprime, cop)
74    IN  [node ↦ {u, v, uprime, vprime},
75         edge ↦ {[from ↦ u, to ↦ v, cop ↦ cop],
76                 [from ↦ u, to ↦ uprime, cop ↦ copprime],
77                 [from ↦ v, to ↦ vprime, cop ↦ copprime2cop],
78                 [from ↦ uprime, to ↦ vprime, cop ↦ cop2copprime]}]
79
```

* Modification History
* Last modified *Wed Jan* 09 16:00:02 *CST* 2019 by *hengxin*
* Created *Wed Dec* 19 18:15:25 *CST* 2018 by *hengxin*