

```

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( $G$ )  $\triangleq$  | A state space is a digraph with labeled edges. |
9     $\wedge$  IsGraph( $G$ ) | It is a digraph (represented by a record). |
10    $\wedge$   $G.node \subseteq (\text{SUBSET } \textit{Oid})$  | Each node represents a document state, i.e., a set of Oid. |
11    $\wedge$   $G.edge \subseteq [\textit{from} : G.node, \textit{to} : G.node, \textit{cop} : \textit{Cop}]$  | Each edge is labeled with a Cop operation. |
13 EmptySS  $\triangleq$  EmptyGraph
14 |-----|
15 Locate( $\textit{cop}$ ,  $\textit{ss}$ )  $\triangleq$  | Locate the (unique) node in state space  $\textit{ss}$  that matches the context  $\textit{ctx}$  of  $\textit{cop}$ . |
16   CHOOSE  $n \in \textit{ss}.node : n = \textit{cop}.ctx$ 
18 xFormSS( $\textit{cop}$ ,  $\textit{copprime}$ )  $\triangleq$  | Transform  $\textit{cop}$  against  $\textit{copprime}$  on state space. |
19   LET  $u \triangleq \textit{cop}.ctx$  | Return the extra state space. |
20    $v \triangleq u \cup \{\textit{cop}.oid\}$ 
21    $u_{prime} \triangleq u \cup \{\textit{copprime}.oid\}$ 
22    $v_{prime} \triangleq u \cup \{\textit{cop}.oid, \textit{copprime}.oid\}$ 
23    $\textit{cop2copprime} \triangleq \textit{COT}(\textit{cop}, \textit{copprime})$ 
24    $\textit{copprime2cop} \triangleq \textit{COT}(\textit{copprime}, \textit{cop})$ 
25   IN  $[node \mapsto \{u, v, u_{prime}, v_{prime}\},$ 
26      $edge \mapsto \{[\textit{from} \mapsto u, \textit{to} \mapsto v, \textit{cop} \mapsto \textit{cop}],$ 
27        $[\textit{from} \mapsto u, \textit{to} \mapsto u_{prime}, \textit{cop} \mapsto \textit{copprime}],$ 
28        $[\textit{from} \mapsto v, \textit{to} \mapsto v_{prime}, \textit{cop} \mapsto \textit{copprime2cop}],$ 
29        $[\textit{from} \mapsto u_{prime}, \textit{to} \mapsto v_{prime}, \textit{cop} \mapsto \textit{cop2copprime}]]]$ 
31 xFormCopCopsSS( $\textit{cop}$ ,  $\textit{cops}$ )  $\triangleq$  | Transform  $\textit{cop}$  against  $\textit{cops}$  (a sequence of Cop) on state space. |
32   LET RECURSIVE xFormCopCopsSSHHelper( $-, -, -$ ) | Return the extra state space. |
33   xFormCopCopsSSHHelper( $\textit{coph}$ ,  $\textit{copsh}$ ,  $\textit{xss}$ )  $\triangleq$  |  $\textit{xss}$ : the eXtra state space |
34     LET  $u \triangleq \textit{coph}.ctx$ 
35      $v \triangleq u \cup \{\textit{coph}.oid\}$ 
36      $uvSS \triangleq [node \mapsto \{u, v\}, edge \mapsto \{[\textit{from} \mapsto u, \textit{to} \mapsto v, \textit{cop} \mapsto \textit{coph}]]]$ 
37     IN IF  $\textit{copsh} = \langle \rangle$  THEN  $[\textit{lss} \mapsto uvSS, \textit{xss} \mapsto \textit{xss} \oplus uvSS]$ 
38     ELSE LET  $\textit{copprimeh} \triangleq \textit{Head}(\textit{copsh})$ 
39        $u_{prime} \triangleq u \cup \{\textit{copprimeh}.oid\}$ 
40        $v_{prime} \triangleq u \cup \{\textit{coph}.oid, \textit{copprimeh}.oid\}$ 
41        $\textit{coph2copprimeh} \triangleq \textit{COT}(\textit{coph}, \textit{copprimeh})$ 
42        $\textit{copprimeh2coph} \triangleq \textit{COT}(\textit{copprimeh}, \textit{coph})$ 
43       IN xFormCopCopsSSHHelper( $\textit{coph2copprimeh}$ ,  $\textit{Tail}(\textit{copsh})$ ,
44          $\textit{xss} \oplus [node \mapsto \{u, v\},$ 
45            $edge \mapsto \{[\textit{from} \mapsto u, \textit{to} \mapsto v, \textit{cop} \mapsto \textit{coph}],$ 
46              $[\textit{from} \mapsto u, \textit{to} \mapsto u_{prime}, \textit{cop} \mapsto \textit{copprimeh}],$ 
47              $[\textit{from} \mapsto v, \textit{to} \mapsto v_{prime}, \textit{cop} \mapsto \textit{copprimeh2coph}]]]$ )
48     IN xFormCopCopsSSHHelper( $\textit{cop}$ ,  $\textit{cops}$ , EmptySS)

```

49 |
|
| * Modification History
| * Last modified Sun *Jan 06 16:57:10 CST 2019* by *hengxin*
| * Created *Wed Dec 19 18:15:25 CST 2018* by *hengxin*